mostly drawings of airports these days...

fault lines — a cultural heritage of misaligned...

fault lines — a cultural heritage of misaligned expectations

Last week I had the pleasure of speaking at ***MuseumNext Melbourne*** `https://www.museumnext.com/speakers/aaron-straup-cope/?event=165` *. I did a talk that I've been threatening to do for a few years now so I was grateful for the chance to work through and better articulate my argument. A number of people said they enjoyed the talk which is always nice and I heard stories of people debating and discussing the talk on their own afterwards, which is even better.*

Mostly for my own benefit of remembering I've included the talk proposal below:

Have museums, and in particular art museums, become too just a little bit too cozy with artists, their estates and their representative groups? Has it always been this way and is the only change that the increasing presence of in-house digital technologies in the a museum context only serves to highlight and reinforce this dynamic? What does it mean for a museum to try and cultivate a meaningful digital practice for their collections and exhibitions when the usage rights for their core assets are held by a third party?

How does the nearly ubiquitous presence so-called "bring-your-own" digital technologies in a museum context affect the issue? How do the abilities and expectations these technologies confer on visitors affect any middle ground that may have previously been established between the rights and demands of artists and the access and preservation goals of a museum?

What does it mean when artists themselves employ these same digital technologies and exploit audience expectations to create their own bespoke museums?

This presentation will plunge in to the topic, look around with a critical eye and endeavour to propose a practical and conceptual framework for where the museum sector goes from here.

This is what I actually said:

Hi, my name is Aaron. The story of my relationship with the cultural heritage sector is fiddly at best and boring at worst so I will just say that while I don't presently work at a museum I still use the second-person plural. I self-identify as "we".

I'd like to start with a little bit of audience participation.

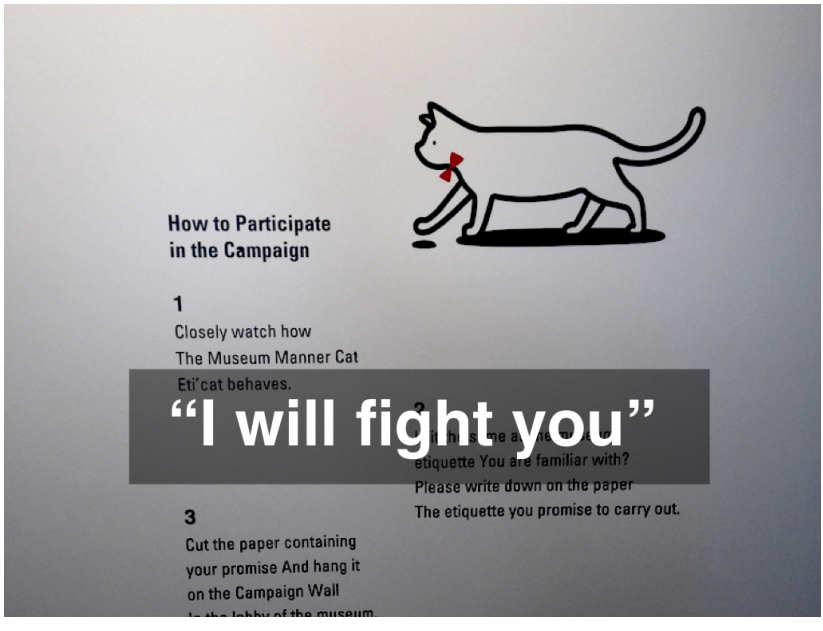
Raise your hand if you've ever been to a museum. Keep your hand raised if you've ever taken a photograph of a wall label to remember something you've seen while visiting a museum. Keep your hand raised, still, if you think that's working out well for you.

Between 2012 and 2015 I was part of **the team at the Cooper Hewitt** `https://labs.cooperhewitt.org/` that built **the Pen** `https://collection.cooperhewitt.org/pen/` . I used to

ask those same three questions during the time we were making the Pen, as a way to explain *why* we were making the Pen. The punch-line was always: Imagine if you never had to do that again? Imagine if you could come to museum with the confidence that remembering your visit would be easy and unintrusive and not require silly workarounds like photographing wall labels. Imagine if you could take for granted that your museum visit wouldn't be defined by all the things you had to do to remember your visit in the first place.

These days I am **working on maps** `https://www.mapzen.com/` again. Specifically, I am working on a project to build **a gazetteer of places** `https://whosonfirst.mapzen.com/` with an open data license and global coverage spanning continents all the way down to neighbourhoods and venues. Every place in the gazetteer has a stable and permanent ID and by extension a stable and permanent URL.

When people ask "why" I sometimes like to say that it's mostly so that we can stop arguing about how to spell place names. What if, instead, we could take for granted that each place had a stable referent off of which we might hang *all the names and all the variations in spelling, in all the languages*? That might be useful if only in that it would allow us to focus on other more important things.



I mention these stories because I want to start by laying my cards on the table for what will follow in this talk. That is: I fundamentally believe that the distinction between museums and libraries and archives, in the minds of people *outside* the cultural heritage sector is collapsing. Assuming they ever thought those distinctions existed in the first place.

There are some people inside the sector who share my opinion but as often as not it is an idea that is met with outright hostility. These are fighting words, to many.

I have sometimes been accused of hyperbole or of not sufficiently understanding (or at least appreciating) the differing roles and responsibilities and historical contexts in which each practice has evolved. Both are fair criticisms but the problem I have with either is that that, whether or not they are true, they don't really address the actual argument I am advancing.

What if all the accussations on both side of the argument are correct?

Perhaps what I am seeing are shadows at dusk — ill-defined and lacking clarity — but that doesn't mean they aren't still there. So what exactly do I mean what I say that the distinction between these three practices has, or is, collapsing?

I mean to suggest that the functions, the external expectations of competencies, of any one professional class are blurring with the others in people's minds.

Why shouldn't a museum have a robust and well-structured body of searchable metadata not just of its collection but also all the ten-thousand word essays that have been written about it? Why shouldn't libraries be able to accept self-deposits both as an intellectual and an operational prerogative? Why shouldn't an archive offer interpretive guidance on the materials they house?

It is a challenge to explain to people outside the sector what actually distinguishes a museum from an archive when the former has storage facilities full of stuff that they can't, or won't, show to people because those objects haven't been catalogued properly. It is doubly challenging when you remember that libraries manage to make sense from a similar, often greater, chaos. It's not as though librarians actually read all the books they keep on hand but still they do enough to foster and cultivate a culture of curiousity and to promote learning and discovery in their patrons.

I think one of the reasons my argument is met with such hostility is that it is predicated on some still nascent changes that contemporary life has afforded us and the practice of preservation has not usually been in the business of nascent technologies.

By changes I mean the Internet, as a whole, and more specifically the permanent (or at least durable) and asynchronous network of documents we call the World Wide Web. This network exists in contrast to the mono-directional and now **increasingly weaponized television and broadcast**

culture http://www.aaronland.info/weblog/2014/09/11/brand/#dconstruct that many hoped would be relegated to the ashes of the 20th century but which has seemed to **return with a vengeance** http://www.aaronland.info/weblog/2014/10/06/interpretation/#brick .

A couple of years ago Jason Scott, **whom many of you will have seen speak this morning** https://www.museumnext.com/speakers/jason-scott/?event=165 , and I attended a different conference together and he and I were discussing his work, as part of both the **Internet Archive** https://archive.org/ and **Archive Team** http://www.archiveteam.org/ , to pre-emptively save things you never knew you were going to miss on the Internet. I am 100% in support not just of these efforts but also their approach. Whatever missteps the Internet Archive and Archive Team make along the way those who follow in our footsteps will benefit from the willingness of Jason and his peers to bet on the future regardless of how dimly their contemporaries may have looked upon the present.

There is however a weak link in Jason's work.

It is a weak link whose consequences are potentially so catastrophic that unless things have gotten really really really bad we will probably never let them come to pass. It is worth recoginizing

that all of Jason's work is built on the foundational layer we've come to know as the electrical grid. All of Jason's work vanishes when the power goes out.

I mention this because when you consider preservationists as a professional class and when you think about their work in an historical context then you start to realize that all they have ever known, on average, is war and pillaging and looting. As such is again important to recognize that although some mistakes have been made over the years they have otherwise done a remarkable job of keeping stuff safe under genuinely extraordinary cicrumstances.

So it's not crazy to imagine that for preservationists — again as an abstract professional class in an even more abstract historical timeline — the jury might still be out on whether electricity is anything we can depend on yet. I have yet to meet a preservationist who has expressed any kind of existential doubt about the electrical grid but, even just as an exercise, it is a possibility worth contemplating.

It seems only prudent.

I share this story because the larger argument I am trying to make in this talk rests on an even more recent and potentially less certain foundation. My argument depends not just on the electrical grid and not just a globally linked network of documents but also on a network of globally linked databases and a layer of applications that we are building on top of that.

The scale and the speed with which computerized and networked databases have shaped contemporary life often lend them

and air and a weight of inevitability that is as confortable as it is misleading.

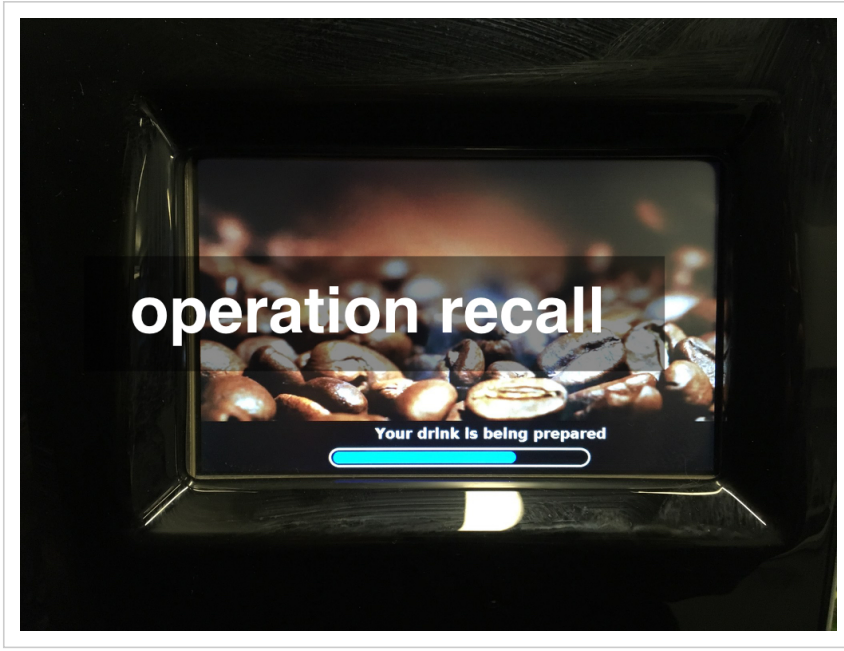
I want to acknowledge that at least one potential flaw in my argument is a misplaced confidence in our shared effort to ensure that we will be able to take for granted these layers of communal infrastructure. We treat these things as natural laws, rather than the shared and concerted efforts they are, at our own peril.

Again, even simply as an exercise, we would do well to imagine a world *without* an Internet. Or even just an Internet changed in nature beyond recognition. But if those possibilities are so terrible to ever let happen then we also need to think about what

they makes possible. We need to to think about how those possibilities change what people expect as commonplace and, by now you might be starting to see a theme emerge, what they might take for granted.

I also want to mention that there is an entirely other talk about the roles and responsibilities that the cultural heritage sector should assume in not simply preserving this network infrastructure but actively running and maintaining it in the service of cultural heritage itself.

But this is not that talk.



Instead this a talk about the operationalization of recall. This is a talk about how recall — the core of what we champion and celebrate as so-called memory institutions — is being normalized by the network.

More than that, even, what I see is an increasing de-fetishization of recall. More and more we take recall for granted in that way we charge the most important aspects of our life with being mundane and unseen. Recall has joined the list of things that are only noticed in their absence.

I think this is a good thing but I worry that the cultural heritage sector, and in particular art museums, is structurally unprepared to adapt to it.

Here again we enter the territory of shadows at dusk. I do not want to suggest that either the problems I am describing, or their remedies, are universal. Like most things in the cultural heritage sector there will be many shades or grey that do not lend themselves easily to an eight-point strategy document. I do mean to suggest that there is something going bump in the night and it is worth our investigating.

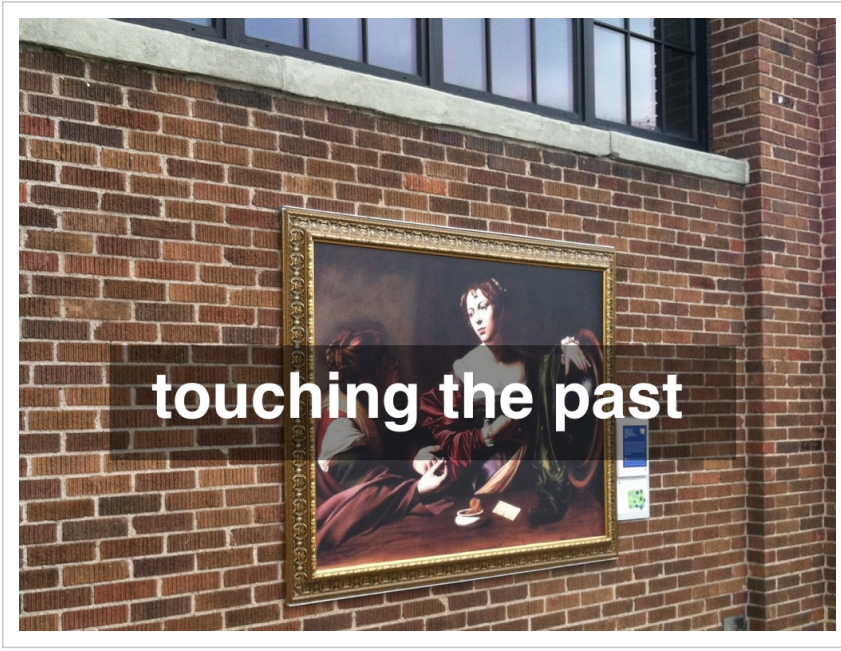
I want to call attention to the assumption of a shared communal network infrastructure *as a public good* that is central to

my argument. It is an assumption that, given the politics of the late 20-teens, no longer seems self-evident. And I also want to recognize that mine is an argument that presumes conditions which may be antithetical on both material and practical levels to the very practice of preservation itself. Maybe.

In the meantime it is hard to deny that we live in a world where reaching out and "touching the sky" is as much about touching the past as it is the present. And that we take this practice for granted.

For every Amazon order or dispatching of a car service or status update there is a Wikipedia query or someone consulting an email archive or a clown on social media being fact-checked. The list goes on. This is not simply about "greater access" to an ever-growing pool of information. More than that it is the ability to take for granted that the past is proximate, in a manner that is genuinely unprecented.

We use the network to circumvent the present and the moment and that's a curious environment for museums to operate in since being present in the moment is largely how we've come to see and to define ourselves.



So, in 2017 what is a museum (or a library or an archive) that does not usefully exist beyond the borders of the moment, beyond the borders of its walls? More importantly what is a museum, in 2017, that can not exist beyond its physical walls because it lacks permission to do so?

What follows is not a comprehensive catalog of ways this happens in 2017, nor are they examples specific to the network. Indeed we, in this room, could spend the rest of the afternoon

documenting ways in which museums are prevented from doing their work. These examples serve only to illustrate the problem:

- *A prohibition on photography in the galleries or an overly heavy-handed response to visitors publishing photos online. A million years ago I worked at Flickr and we were constantly fielding take-down notices from the various French artists rights associations about tourist-quality photographs taken in the galleries.*

- *Perpetuating the madness around "print quality" images in a universe where the minimum required image size for a musuem's iPad app exceeds the minimum required image size for a print publication.*

- *Limiting or restricting catalog records from being published online because are not perfect or because they might "upset" someone.*

- *Loan objects never being mentioned or included in a catalog or, worse, being removed after an exhibition comes down. This one is especially galling to me since we tell people that it's very important that they come see these objects and then pretend as though it never happened.*

- *Generally limiting curatorial authority or decision making, whether it's a traveling exhibition or a retrospective of a contemporary artist's work. It's a practice that is fine if you're David Hockney and you want to hold a major retrospective of your own work at the Royal Academy which is basically a private members club. But that exhibition, in 2012, perhaps more than any other in recent memory* ***highlighted the need for and the purpose of*** `http://www.aaronland.info/weblog/2012/02/14/incentivize/#shadows` *curatorial discretion.*

What these examples point to is an imbalance in the relationship between museums and their dancing partners. That imbalance is further reflected in the inability to meaningfully interact on or with the network.

It reflects an inability for museums to engage with the network because of an overzealous regiment of permissioning or to use the network as a tool by which, **to borrow Elaine Gurian's phrase** `https://www.museumnext.com/speakers/elaine-heumann-gurian/?event=165` , they might "promote nuance" and a more complex understanding of their collections.

That is also some pretty fancy talk for a pretty simple idea: That is it time for the cultural heritage sector, as a whole, to pick a fight.

It is time for the sector to pick a fight with artists, and artist's estates and even your donors. It is time for the sector to pick a fight with anyone that is preventing you from being allowed to have a greater — and I want to stress greater, not total — license of interpretation over the works which you are charged with nurturing and caring for.

The following passage did not make it in to the talk which was only 20 minutes long but since we have the luxury of time and space here: This has been a curious position to arrive at. If I studied anything it was painting and studio arts so I am broadly sympathetic with the demands of artists to maintain control and ownership of their work. I also closely followed and supported the efforts in the comix world leading up to the ***Creator's Bill of Rights*** `https://en.wikipedia.org/wiki/Creator%27s_Bill_of_Rights` *which is an industry pretty much defined by how poorly publishers have treated artists and writers. There are good reasons why we've ended up here but I also feel as though we've gone from one extreme to the other which counts as dubious progress in my book.*

It is time to pick a fight because, at least on bad days, I might even suggest that the sector has been played. **We all want to outlast the present** `https://twitter.com/auchmill/status/832441248157872129` , and this is especially true of artists. Museums and libraries and archives are **a pretty good bet** `https://museumvictoria.com.au/discoverycentre/infosheets/sam-the-koala/` if that's your goal.

Consider the number of works sent to MoMA, in the 1970's, after one of their registrars let it be known he would accession in to the collection anything that arrived on his desk by mail. Consider the

well-known tactic of self-depositing any book with an ISBN number to the Library of Congress. If you do they are required to care and feed for that book for the rest of forever. Consider how happy people were to learn that the Library of Congress had acquired all the Twitter messages, and with it their tiny contribution, for precisely that reason.

Consider the growing number of artists and private companies who are creating their own museums in the services of... themselves.

It remains to be seen how well those last institutions will fare over time. Indeed being "in it for the long-run" and in being wired for the long-run, both operationally and intellectually, is what distinguishes the cultural heritage sector from other endeavours and the public is made better by those efforts.

My argument though is that taking on the burden of championing and preserving cultural heritage is a two-way street. My argument is that the sector has become prone to an unhealthy deference, to being little more than caretakers in the service of

someone else's enterprise that is legitimized by our labours. There exists an imbalance in our relationship with those whose works we shepherd that, at best, hampers our ability to actively participate with the network and for our own work to square with people's expectations of what is possible and what can be taken for granted.

Ask yourselves: Can I actually *do* anything with my museum's collection sitting here in this or any other random conference room, right now? I suspect that that for most, probably all, of you the answer is resolutely: No, or at least not well. Those few ways in which it is possible for our collections to participate with the network are still surface-level, at most. Now ask yourselves why that is.

In closing I would like to leave you with an open question, perhaps even a provocation: Does the cultural heritage sector's current fascination with crafting "experiences" in fact betray our anxieties about the network? Or put another way: Is the vogue for fashioning these all-consuming, in-gallery shock-and-awe immersions actually a way for us to distract our visitors from the reality that there is often no recall of any consequence of that experience the minute they walk out the door?


YOUR INTERNET REINVENTED
That's actually pretty badass.
thank you

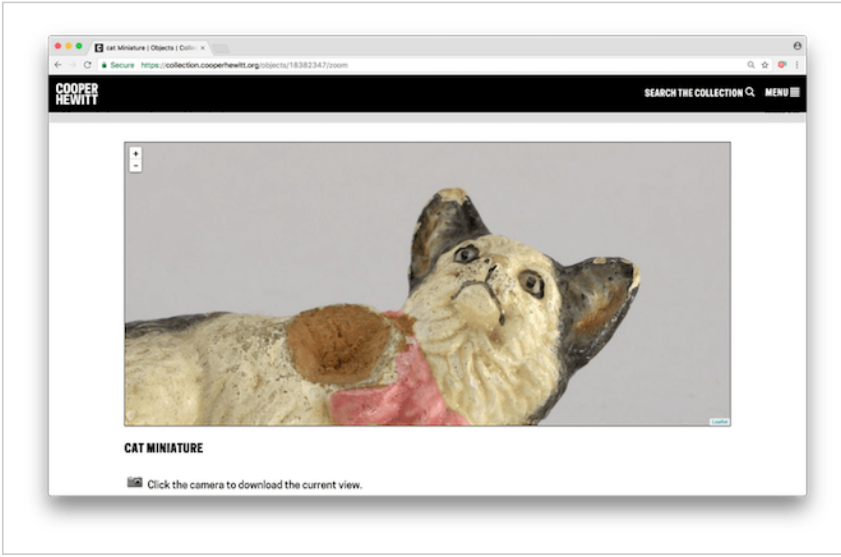
Thank you.

2017-02-20

numbers for the record

go-iiif, in production

go-iiif, in production



On Friday Micah **flipped the switch to enable zoomable images** `https://labs.cooperhewitt.org/2017/parting-gifts/` on the collections website at the **Cooper Hewitt** `https://collection.cooperhewitt.org/` . Despite not being employed by the museum anymore this is a project I've been helping out with during the margins of the day. It's been a good way to test a bunch of choices and decisions I've made around the **go-iiif** `http://www.aaronland.info/weblog/2016/09/18/marshmallows/#iiif` project that I started last year.

Rather than deploying an **on-the-fly IIIF server** `http://iiif.io/api/image/2.1/` all of the zoomable-image tiles were pre-rendered. What follows are copies of two emails I posted on the **iiif-discuss mailing list** `https://groups.google.com/forum/#!searchin/iiif-discuss/Cope%7Csort:relevance/iiif-discuss/uI5k_3NsYv4/MwlWUSEDBgAJ` , describing some of the nuts and bolts of tiling all those images, and then a third email sent privately explaining why to pre-tile instead of running a production server.

I just finished tiling a large museum collection that was recently digitized in its entirety at a minimum of 4096 pixels per side. I did this to test a bunch of questions I had/have about IIIF as well as **the go-iiif implementation that I wrote** `https://thisisaaronland.github.io/go-iiif/` to investigate those questions.

Here are the numbers, passed along as an FYI:

- *289, 208 images – as measured by the number of "info.json" files produced*
- *58, 977, 784 tiles produced (for scale factors 8,4,2,1)*
- *518 GB of (tiles) data*
- *45 days processing / transfer time (averaging out to ~ 12 seconds per tile)*

The bottleneck in the process was always CPU coupled with disk I/O initially (when writing to a local disk) and then some amount of network transfer once we started writing directly to S3.

To be honest, by the time we got around to writing directly to S3 I was less interested in endless performance tweaks (this was a mornings and

weekends project) and reasonably certain that any non-CPU related improvements would be incidental since the real problem is that crunching pixels remains CPU-expensive.

Over the course of the 45 days, using an 8 core machine with 30GB RAM, all of the CPUs were pegged at 80-100% usage and memory usage never went above ~ 8GB. CPU cost was probably exacerbated by the (seeming ?) inability of **the image processing library** `https://github.com/h2non/bimg/` to return a new smaller image when cropping the source. Dunno - mornings and weekends, right :-)

To put the 518 GB of tile data in perspective, that's 518 GB worth of bytes which doesn't take in to account filesystem specifics like block size or inode count. For example before switching to S3 we had filled up 619 GB on a 1 TB disk using 40% of the inodes (ext-4 with a block size of 2048 bytes) and had tiled less than 50% of the total images.

Finally, when writing directly to disk on a EC2 machine disk I/O was brutal to put it mildly. The software had to be repeated throttled to account for the fact that AWS would limit writes (on this particular instance type) to about 120MB/second.

And then the follow-up to that email:

A quick follow-up based on some comments I've received about this post, specifically about the time (45 days) to process all the images:

The bottleneck as I mentioned was some combination of disk I/O, network and CPU. AWS aggressively throttles I/O on EC2 machines so there's not much to do there. Bandwidth inside AWS (from EC2 to S3) is pretty fast but again I expect that is throttled too. The CPU issue is an open question. Or more specifically CPU and a gazillion concurrent processes.

The ***libvips*** `http://www.vips.ecs.soton.ac.uk/index.php?title=VIPS` *author mentioned that he's written a "deep zoom" tiling client that is impressively fast. It's entirely possible that some combination of the Go bindings and my code on top of that have introduced problems. I would never rule that out, especially for a "mornings and weekends" project.*

Like I said, it's not clear to me how/whether the generic libvips API allows to crop an existing image without also transforming it, meaning that every crop requires processing the same bag of original pixels (even if you keep them hanging around in memory).

The other thing I didn't mention in my note to the mailing list is that I introduced some very aggressive blocking throttles in my code mostly in an effort to keep the OS from freaking out and imposing its own limits or simply grinding to a halt. These limits were pretty arbitrary and once they achieved something approaching "good enough is perfect" I just let things run their course.

I expect that accounts for a meaningful portion of the 45 days but more testing would be good, for sure, even if it was just spreading the load across (n) machines in parallel.

Finally, it seems pretty clear to me that the pre-tiling IIIF images problem has "**please write an AWS Lambda function** `http://docs.aws.amazon.com/lambda/latest/dg/getting-started.html` " written all over it. We can debate the relative merits of abstracting general purpose computing further and further in to commercial services but for a small library or museum having the ability to put a bag of images in one S3 bucket and generate another bag of (tiled) images in another S3 bucket for a one-time fixed cost is pretty attractive.

To be clear: I have no idea what it would cost to tile 280K images using a Lambda function (and **the PUT costs in S3** `https://aws.amazon.com/s3/pricing/` will

> start to add up... like 58M times) but maybe it's still less that compute time on a big EC2 machine and certainly not having to deal with the hassle of things like "installing libvips" is worth time and money, especially for institutions with limited staff.

Speaking of which, since publishing this post I've learned that ***Roger Howard has written an AWS Lambda*** `https://github.com/rogerhoward/lambdazoom` *function which "converts uploaded images to the Deep Zoom tiled image format" so that's quite good.*

And then finally this email, which was sent privately, on the rationale for pre-tiling over running a live server against production traffic (read: strangers on the Internet):

I would not run *any* IIIF server live in front of production traffic, **including mine** https://github.com/thisisaaronland/go-iiif#iiif-server .

I was planning to write a long and twisty museum paper about just this subject but now I am not going to, so I may just write a long and twisty blog post instead.

Before I get in to why you shouldn't run an IIIF server in production I am not sure I completely understand your setup. Are you saying that source image lives on S3? If that's the case then that would explain why "info.json" takes as long as it does to generate.

In order to generate the info.json file you need to know the dimensions of the file itself which means the file needs to be loaded (from whatever its source is) and then again in to memory, so this bit here:

https://github.com/thisisaaronland/go-iiif/blob/master/cmd/iiif-server.go#L232-L255 https://github.com/thisisaaronland/go-iiif/blob/master/cmd/iiif-server.go#L232-L255

If that's the case then I'd start by checking how long it takes to download the image itself from S3 and compare it to the XHR request for info.json. How big is the source image? If the image is being loaded locally that's weird. *Note: It wasn't*

Anyway, and specifically for your stuff, I would pre-cache all the tiles up front. There are only a few thousand images so it shouldn't take very long. To give you some sense of how long it takes with a larger dataset there's this:

https://groups.google.com/forum/#!topic/iiif-discuss/uI5k_3NsYv4 `https://groups.google.com/forum/#!topic/iiif-discuss/uI5k_3NsYv4`
Note: This is just a link to two email messages transcribed above.

Again, it's not really the total number of images that's the problem so much as how many any one class of EC2 machine can handle at once.

*I also may have downloaded **all 250GB worth of the Met's public domain images*** `https://github.com/thisisaaronland/openaccess` *recently and plan to test things again with their dataset...*

The issue with pre-tiling will be I/O and CPU contention but again for a small dataset it's not a big

deal.

Pre-cached tiles + Leaflet + **Jack Reed's thing to deal with the IIIF nonsense** `https://github.com/mejackreed/Leaflet-IIIF` is great because it's simple and easy and pretty much bog-standard not-bleeding-edge JavaScript that has a hope of lasting more than 6 months before it needs to be fixed. And doing incremental updates to images as they change is trivial.

Running a IIIF tiling server in production is kind of madness though because of the I/O contention and the fact that any given server has a finite amount of RAM.

If you implement the spec without thinking about server-side caching then you end up loading the same multi-MB-sometimes-even-GB file in to memory for every 256 square pixel tile that is requested and doing some fairly CPU intensive pixel-crunching for each one, all at the same time. With a scenario like that it's only a question of how soon your server will melt down and stop being able to serve traffic.

Even if you cache source images aggressively in memory eventually you will run out of memory, given enough traffic.

The go-iiif server (and friends) were a way to work through these concerns and see if maybe I was missing something. I don't think I am. A live IIIF server is great for you and your 4 closest librarian friends but actually scaling this thing in any meaningful way (without just burning cash on hardware) appears to have been an afterthought.

Anyway, if your source images are on S3 then you can get the iiif-tile-seed program to fetch them by defining your "source" config like this:

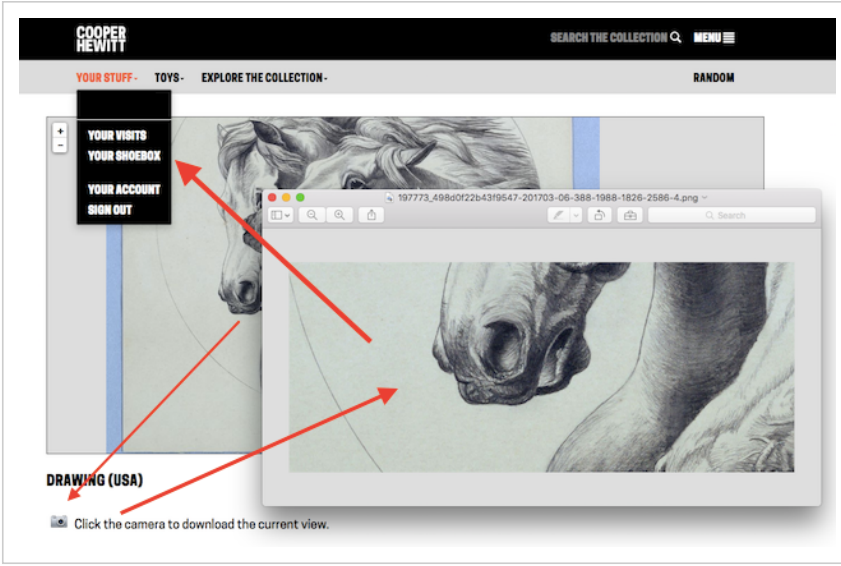
https://github.com/thisisaaronland/go-iiif#s3 `https://github.com/thisisaaronland/go-iiif#s3`

And you can write the tiled images _back_ to S3 by defining your "derivatives" config like this:

https://github.com/thisisaaronland/go-iiif#s3-1 `https://github.com/thisisaaronland/go-iiif#s3-1`

Just make you write the tiles to a different or child directory of the source directory. If you don't then AWS will happily overwrite your original source files with the directory (which shares the same names as the original file) containing your derivatives. Because [redacted].

But anyway, **zoomable images** `https://collection.cooperhewitt.org/objects/18382347/zoom` ! One of the nice things about the zoomable images on the Cooper Hewitt website is **the ability to generate and download a static image of the area you've zoomed in on** `https://github.com/mapbox/leaflet-image` . This was something I tried to get included in **the scope for the interactive tables** `http://www.aaronland.info/weblog/2015/04/10/things/#mw2015` when the museum reopened, but it never made the cut which is too bad. We had everything in place to allow you to upload those crops to **your visit pages** `https://labs.cooperhewitt.org/2015/exporting-your-visits/` and everything would have been properly cross-referenced with the original object.



That activity would have also been a valuable **"metric" for "user-engagement"** `https://collection.cooperhewitt.org/stats` (...we'll save a longer discussion of that "subject" for another day). People save and collect things for later for all kinds of reasons, many of them absent-minded, but taking the trouble to crop and save a detail requires a certain amount of deliberation. The temptation to data-mine those image crops would have been strong but to do so would be creepy so just knowing which images had been cropped and saved, and how many times, seems like a reasonable compromise.

I don't know whether that feature will ever get built in to the tables. It should, and if any museum is **in a position to void the warranty, with confidence, on their third-party work** `http://www.aaronland.info/weblog/2016/11/04/seealso/#london` it is the Cooper Hewitt. In the meantime because those interactive tables are just consumers of **an API that the museum controls** `http://collection.cooperhewitt.org/api` and because there are already the infrastructure to save objects on the collections website itself there is only a small amount of time-and-typing necessary to implement **saving an image detail to your online "shoebox"** `https://labs.cooperhewitt.org/2015/collect-all-the-things/` .

Maybe the museum will do that. That would be good.

2017-03-05

things I have written about elsewhere #20170404

The Who's On First API

The Who's On First API

*This was **originally published on the Mapzen weblog*** `https://mapzen.com/blog/whosonfirst-api/` *, in April 2017.*

Today we are pleased to announce **the Who's On First API** `https://mapzen.com/documentation/wof/` . This is something that has been hiding in plain sight for a little while now and that a few people may have noticed if they "looked under the hood" of the **Spelunker code** `https://github.com/whosonfirst/whosonfirst-www-spelunker/` following the recent blog post about the **Who's On First bundler tool** `https://mapzen.com/blog/bundler/` .

The API provides programmatic access for you and your robots to all the **Who's On First data** `https://github.com/whosonfirst-data` . You can query

individual
places https://mapzen.com/documentation/wof/methods/#whosonfirstplaces and their relations, look for
concordances https://mapzen.com/documentation/wof/methods/#whosonfirstconcordances and perform **basic spatial**
queries https://mapzen.com/documentation/wof/methods/#whosonfirst.places.getIntersects . You can also use the API to query things like all the different
placetypes https://mapzen.com/documentation/wof/methods/#whosonfirstplacetypes or
sources https://mapzen.com/documentation/wof/methods/#whosonfirstsources for data. It is not a "complete" API yet. The first goal for the API is to achieve parity with the
Spelunker https://whosonfirst.mapzen.com/spelunker/ : anything you can do by clicking around that website manually should be able to be automated using code.

You should treat this API as though it were still in “beta”. Which is to say: the point is for *the thing to work* but there are probably still some rough edges and lingering gotchas so you should adjust your expectations and your code accordingly. In the meantime have at it and **please let us know on**
Twitter https://twitter.com/alloftheplaces or **contact us through email** mailto:hello@mapzen.com if something is busted or just doesn’t feel right.

If you want to dive in right away and come back for storytime later the documentation is here:

https://mapzen.com/documentation/wof/ `https://mapzen.com/documentation/wof/`

"Stuff over HTTP"

The API uses a parameter-based **stuff over HTTP** `http://www.aaronland.info/weblog/2014/02/10/36hours/#what` style interface. Currently all API methods are read-only so everything is sent using **the HTTP GET verb** `https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.3` . If the method is successful the result with be returned with an HTTP `200 OK` response. If there was a problem with the request then an error will be returned in **the HTTP 400-499 range** `https://mapzen.com/documentation/wof/errors/#client-side-errors` . If there was a problem generating a response then an error will be returned in **the HTTP 500-599 range** `https://mapzen.com/documentation/wof/errors/#server-side-errors` .

Successful API responses can be returned in one of three possible formats:

JSON `https://mapzen.com/documentation/wof/formats/#json` ,

CSV https://mapzen.com/documentation/wof/formats/#csv and Who's On First's own "**meta** https://mapzen.com/documentation/wof/formats/#meta " format (which is just a CSV file with fixed headers). JSON is the default response format and available for all methods. CSV is available for most methods. Meta responses are available for methods where it makes sense.

Meta files are the CSV files included with every **Who's On First data repository** https://github.com/whosonfirst-data . The easiest way to think about meta files is that they canned indices or views in to the data; a way for people to grab or work with a slice of the data, say by placetype, without having to set up and load everything in to a database. As such, a lot of tooling has been built to consume meta files from the **I Am Here** https://mapzen.com/blog/iamhere tool, to the Who's On First **point-in-polygon server** https://github.com/whosonfirst/go-whosonfirst-pip/ , to the tools we use to **generate bundles** https://github.com/whosonfirst/go-whosonfirst-clone#example (you can read more about bundles **over here** https://whosonfirst.mapzen.com/bundles/), to indexing WOF data in **in third-party tools like Tile38** https://github.com/whosonfirst/go-whosonfirst-tile38#example-1 to **generating FeatureCollections** https://github.com/whosonfirst/py-

mapzen-whosonfirst-utils/blob/master/scripts/wof-csv-to-feature-collection . Now rather than just consuming the **default meta files** https://github.com/whosonfirst-data/whosonfirst-data/tree/master/meta included with each repository all of those tools can be fed the output of whatever query you dream up using the **whosonfirst.places.search** https://mapzen.com/documentation/wof/methods/#whosonfirst.places.search , or similar, API methods.

Other API responses can and will be added as time and circumstance permit. We are interested in people *using* the Who's On First API in whatever format their software needs and with not in making our opinions about API responses someone else's problem. If there's a particular response format you need to start using the Who's On First API please send up a flare **via**

email `mailto:hello@mapzen.com` or

Twitter `https://twitter.com/alloftheplaces` .

Examples

whosonfirst.concordances.getById `https://mapzen.com/documentation/wof/methods/#whosonfirst.concordances.getById`

Here are all the concordances in Who's On First for the **GeoPlanet** `https://whosonfirst.mapzen.com/spelunker/concordances/geoplanet/` ID for the city of **Montreal** `https://whosonfirst.mapzen.com/spelunker/101736545` or `gp:id=3534`:

```
curl -s -X GET 'https://whosonfirst-api.mapzen.com?method=whosonfirst.concordan
{
    "concordances": [
        {
            "dbp:id": "Montreal",
            "fb:id": "en.montreal",
            "fct:id": "03c06bce-8f76-11e1-848f-cfd5bf3ef515",
            "gn:id": 6077243,
            "gp:id": 3534,
            "loc:id": "n80132975",
            "nyt:id": "N59179828586486930801",
            "qs:id": "239659",
            "tgn:id": "7013051",
            "wd:id": "Q340",
            "wk:page": "Montreal",
            "wof:id": 101736545
        }
    ],
    "cursor": null,
```

```
    "next_query": null,
    "page": 1,
    "pages": 1,
    "per_page": 100,
    "stat": "ok",
    "total": 1
}
```

whosonfirst.places.getByLatLon `https://mapzen.com/documentation/wof/methods/#whosonfirst.places.getByLatLon`

Here are all the neighbourhoods (there's only one of them) at the corner of **16th and Mission** `https://whosonfirst.mapzen.com/iamhere/#17/37.76494/-122.41944` in San Francisco:

```
curl -s -X GET 'https://whosonfirst-api.mapzen.com?method=whosonfirst.places.ge
{
    "places": [
        {
            "geom:latitude": 37.758768,
            "geom:longitude": -122.413313,
            "mz:uri": "https://whosonfirst.mapzen.com/data/858/874/43/85887443.
            "wof:country": "US",
            "wof:id": 85887443,
            "wof:name": "Mission District",
            "wof:parent_id": "85922583",
            "wof:placetype": "neighbourhood",
            "wof:repo": "whosonfirst-data"
        }
    ],
    "stat": "ok"
}
```

There are a couple things to note about this API response. First, the "**minimum response data** `http://code.flickr.net/2008/08/19/standard-photos-response-apis-for-civilized-age/` " for a place and second the use of the `extras` parameter to include additional information about a place. We're still working out what the final "minimum response" data structure for a place should be but so far we've settled on the following properties:

```
{
    "wof:id": ...,
    "wof:name": "...",
    "wof:parent_id": ...,
    "wof:placetype": "...",
    "wof:repo": "..."
}
```

Every API method that returns a "place" as part of its response will be guaranteed to include those properties. If you need or want additional properties you can list them in the `extras` parameter. Passing the `extras=geom:latitude,geom:longitude` parameter will cause the API to add the geographic center a place's geometry to its response. You can also request entire classes of properties by passing only a prefix.

whosonfirst.places.getInfo `https://mapzen.com/documentation/wof/methods/#whosonfirst.places.getInf`

o

For example to fetch **all the names** `https://mapzen.com/blog/wikipedia-data/` for the city of Beijing you would pass `extras=name:`, like this:

```
curl -s -X GET 'https://whosonfirst-api.mapzen.com?method=whosonfirst.places.ge
{
    "place": {
        "name:ace_x_preferred": [
            "Beijing"
        ],
        "name:ady_x_preferred": [
            "\u041f\u0435\u043a\u0438\u043d"
        ],
        "name:afr_x_preferred": [
            "Beijing"
        ],
        "name:als_x_preferred": [
            "Peking"
        ],
        "name:amh_x_preferred": [
            "\u1264\u12ea\u1302\u1295\u130d"
        ],
        "name:ang_x_preferred": [
            "Beicing"
        ],

    ...truncated for brevity...

    "name:zho_x_preferred": [
            "\u5317\u4eac\u5e02"
        ],
        "name:zho_x_variant": [
            "\u5317\u4eac\u5e02"
```

You can request any fully-qualified property, or property prefix, in the `extras` parameter. The API will check to ensure that

the prefix for an extras parameter is defined in the **whosonfirst-sources** `https://github.com/whosonfirst/whosonfirst-sources/tree/master/sources` list (there are **corresponding API methods for sources** `https://mapzen.com/documentation/wof/methods/#whosonfirstsources`). As of this writing if an unknown prefix is requested it is silently ignored. *That might become an error in time. We'll see...*

whosonfirst.places.search `https://mapzen.com/documentation/wof/methods/#whosonfirst.places.search`

Here are all the **microhoods** `https://whosonfirst.mapzen.com/spelunker/placetypes/microhood` in Who's On First returned as a CSV document:

```
curl -s -i -X GET 'https://whosonfirst-api.mapzen.com?method=whosonfirst.places
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: text/csv
Date: Fri, 31 Mar 2017 21:41:50 GMT
Server: nginx/1.4.6 (Ubuntu)
Status: 200 OK
X-api-format-csv-header: wof_country,wof_id,wof_name,wof_parent_id,wof_placetyp
X-api-pagination-cursor: cXV...c7MDs=
X-api-pagination-next-query: method=whosonfirst.places.search&amp;placetype=mic
X-api-pagination-page:
X-api-pagination-pages: 60
X-api-pagination-per-page: 10
X-api-pagination-total: 595
Content-Length: 746
```

```
Connection: keep-alive

wof_country,wof_id,wof_name,wof_parent_id,wof_placetype,wof_repo
US,1108802091,"Scripps Miramar Ranch",1108802089,microhood,whosonfirst-data
US,1108561153,"La Tuna Canyon",85865489,microhood,whosonfirst-data
US,1108750019,"Green Valley Ranch",420781629,microhood,whosonfirst-data
US,1075806299,"Northridge West",85838305,microhood,whosonfirst-data
DE,1108810201,Biesdorf-Sud,-3,microhood,whosonfirst-data
US,1108719769,"Rivergate Industrial District",85846673,microhood,whosonfirst-da
US,1108750017,Gateway,420781629,microhood,whosonfirst-data
US,1108750051,"Northeast Park Hill",85840547,microhood,whosonfirst-data
US,1091648325,"Hansen Dam",85865477,microhood,whosonfirst-data
US,1041491317,"Beverly Glen",85869119,microhood,whosonfirst-data
```

There are a couple of things to note about CSV formatted responses:

- *The `:` character that normally separates a prefix from its value has been replaced by a `_`. For example `wof:id` becomes `wof_id` when it is encoded as a CSV column header. It's not awesome but lots of tools that consume CSV have problems with column headers containing colons so we just live with the inconsistency.*

- *Nested values (arrays and dictionaries) are returned as JSON-encoded strings. It is left up to consumers of the API to decode them as necessary.*

Also, did you notice the many different pagination properties in the JSON responses and the HTTP headers for the CSV response? Yeah...

A short miserable history of pagination

Pagination, the practice of chunking a lot of results in to smaller sets, shouldn't be complicated. But it is. Because databases, after all these years, are still complicated beasts.

Databases have always been about trade-offs. No two databases are the same and so no two sets of trade-offs are the same either. The really short version is that some databases can't tell you exactly how many results there are for a given query. Some

databases can tell you how many results there are but can't or won't return results past a certain limit. Other databases can do both but only if you use something called a cursor for pagination rather than the traditional offset and limit model (as in "return the next 5 of 50 results starting from postion 20").

A long time ago, **I worked at the photo-sharing website Flickr** `https://www.flickr.com/photos/paulhammond/2883048077/` . One of the limits of the **Flickr API** `https://www.flickr.com/services/api` is that the `flickr.photos.search` method was, and still is, capped at 4,000 results for any single query. This was a limit of **Flickr's search engine** `http://aaronland.info/talks/mw10_machinetags/#92` , an early and proprietary version of applications like **Solr** `https://lucene.apache.org/solr/` and more recently **Elasticsearch** `https://www.elastic.co/guide/en/elasticsearch` which are often referred to as "document stores".

Document stores are similar to relational databases (think of the many different `something-SQL` databases people talk about) in that you can query them and get stuff back. Relational databases and document stores are very different in how they accomplish the same task and one of the limitations of the latter has always been that they aren't really designed to return the "long tail" of results for a query with lots of results.

When I last checked the Flickr website it returns **1,844,154 photos when you search for kittens** `https://www.flickr.com/search/?text=kittens` . If you query the Flickr API for photos tagged "kittens" it says there are 429,203 photos. The disparity is because people are allowed to opt-out of including their photos in API results.

Here is the first result at around the 4,000 mark (500 photos per request * page 8, out of a possible 859):

```
https://api.flickr.com/services/rest/?method=flickr.photos.search&api_key=*****

{ "photos": { "page": 8, "pages": "859", "perpage": "500", "total": "429203",
    "photo": [
          { "id": "33746803785", "owner": "148036032@N07", "secret": "4f91e6622
```

And here's the same query but at the 5,000 mark (500 photos per request * page 10):

```
https://api.flickr.com/services/rest/?method=flickr.photos.search&api_key=*****

{ "photos": { "page": 10, "pages": "859", "perpage": "500", "total": "429203",
    "photo": [
          { "id": "33746803785", "owner": "148036032@N07", "secret": "4f91e6622
```

See what's happening there? It's the same photo at the beginning of the list even though the results are a thousand photos

apart. The search engine is configured to return the same results for every request over the 4,000 results limit. This is not a new thing. This is also not a Flickr thing. It's a document store thing. Even today, ten years, later the state of the art hasn't improved all that much: In 2017, Elasticsearch still has **a hard limit of 10,000 results** `https://www.elastic.co/guide/en/elasticsearch/guide/current/pagination.html` for a single query.

The reason I mention all of this is not to pick on Flickr or Elasticsearch or anyone else working with document stores. The performance of "traditional" relational databases is also known to increasingly degrade as you offset further and further in to a large result set (the rule of thumb is that things start to get painful around the 500,000 record mark). The limits imposed by document stores are often worth it because they allow you perform complex queries that would otherwise be impossible or impractical in another database. I mention all of this for two reasons:

First, there remains a lot of interesting and important work to do designing interfaces and models to account for the inability of databases to reliably (or efficiently) return all the results for a large query. The 4,000 results limit was **well-known by developers** `https://www.flickr.com/groups/api/discuss/72157663968579450/` using the Flickr API. We didn't hide that it happened but we also didn't see it as an opportunity to encourage developers to think about different ways to imagine what search

should look like or how people should think about using it. The work that **George Oates** https://en.wikipedia.org/wiki/George_Oates did redesigning the search and subject pages for the **Internet Archive's Open Library project** https://archive.org/details/Kohacon2010OpenLibraryPresentation-GeorgeOates is a good example of how we might approach that problem, going forward.

Second, the reason for telling you all of this "exciting" detail from the past is to help explain how and why we've implemented pagination for the Who's On First API in the present. Who's On First is all about **the relationship between places** https://whosonfirst.mapzen.com/placetypes and sometimes a place (like a country) will have lots and lots of relationships (like all the venues in that country). We don't really have the luxury of "ranking" those relationships and only returning the top 4,000 (or 10,000) results. We need to find a way to return **all the things** https://mapzen.com/blog/all-of-the-places .

Further, Who's On First uses multiple, different databases by design. We do this to ensure that the data and the overall data modeling is flexible enough to work with as many tools as possible. That's very important to the project because we want to make sure that the infrastructure burden required to *do* something with Who's On First data is as a light as possible and not a reflection of Mapzen's

specific needs to operationalize things or the inertia of our own preferences.

For example the **Spelunker** `https://whosonfirst.mapzen.com/spelunker` is built using Elasticsearch but has no spatial indices even though they are supported. We set things up this way to make sure, to prove, that it was possible to use the data without *requiring* a spatial database.

Since there is no all-purpose database, the Who's On First API accounts for multiple different pagination models. We've identified four overlapping models (**plain** `#pagination-plain`, **cursor** `#pagination-cursor`, **mixed** `#pagination-mixed` and **next-query** `#pagination-next-query`) each of which are described in detail below. If you don't really care and just want to get started **you should skip ahead to the discussion of next-query pagination** `#pagination-next-query`.

Plain pagination

Plain pagination assumes that we know how many results a query yields and that we can fetch any set of results at any given offset. For example, let's say you wanted to use the API to fetch all the places with a variant name containing the word

`Paris` in sets of five. The API will respond with something like this:

```
{
    "places": [ ... ],
    "next_query": "method=whosonfirst.places.search&alt=Paris&per_page=5&page=2
    "total": 7,
    "page": 1,
    "per_page": 5,
    "pages": 2,
    "cursor": null,
    "stat": "ok"
}
```

It's pretty straightforward. There are seven results (`total`) and this is the first of two pages worth of results (`page` and `pages`, respectively). You might already be wondering about the `next_query` property but we'll get to that shortly `#pagination-next-query` .

Cursor-based pagination

Cursor-based pagination is necessary when a database can't or won't tell you how many results there are for a query. This means you will need to pass the same query to the database over and over again for as long as the database returns a

`cursor` which is like a secret hint *that only the database understands* indicating where the next set of results live.

For example, let's say you wanted to use the API to fetch all of the venues near the Smithsonian Cooper Hewitt Design Museum `https://whosonfirst.mapzen.com/spelunker/id/420571601/` in sets of ten. The API will respond with something like this:

```
{
    "places": [ ... ],
    "next_query": "method=whosonfirst.places.getNearby&latitude=40.784165&longi
    "per_page": 10,
    "cursor": {CURSOR},
    "stat": "ok"
}
```

In order to fetch the next set of results you would include a `cursor={CURSOR}` parameter in your request, rather than a `page={PAGE_NUMBER}` parameter like you would with plain pagination. Some databases yield time-sensitive cursors that expire after a number of seconds or minutes so the easiest way to think about cursors is that they are *all* time sensitive.

Databases, amirite?

Mixed pagination

This is where it gets fun. Sometimes an API method might use *both* plain and cursor-based pagination. That can happen when an underlying database is able to calculate the total number of results but only be able to fetch a fraction of them using plain pagination after which it needs to switch to cursor-based pagination. Which doesn't really make any sense when you think about it because cursors are magic database pixie-dust so there's no way to determine or calculate a corresponding cursor for a traditional page number. So in the end the API itself needs to perform an initial query just to see how many results there are and then adjust whether it is going to use plain or cursor-based pagination on the fly.

For example, let's say you wanted to use the API to fetch all the `microhoods` in sets of five. The API will respond with something like this:

```
{
    "places": [ ... ],
    "next_query": "method=whosonfirst.places.search&placetype=microhood&page=2&
    "total": 186,
    "page": 1,
    "per_page": 5,
    "pages": 38,
    "cursor": null,
    "stat": "ok"
```

```
}
```

But if you then asked the API to fetch all of the `neighbourhoods`, again in sets of five, the API will respond with something like this:

```
{
    "places": [ ... ],
    "next_query": "method=whosonfirst.places.search&placetype=neighbourhood&pe
    "total": 81065,
    "page": null,
    "pages": 16213,
    "per_page": 5,
    "cursor": "{CURSOR}",
    "stat": "ok"
}
```

In both examples we know how many results there will be. In the first example we are able to use plain pagination so we know that this is page one of thirty-eight and thus the value of the `cursor` property is null. In the second example the API has returned a cursor so even though we know the total number of results and can calculate the number of "pages" we set the value of the `page` property to be null since the requirement on cursor-based pagination makes it moot.

If you look carefully at the value of the `next_query` property in both examples you can probably figure out where

this is going, next.

Next-query-based pagination

Next-query based pagination is an attempt to hide most of the implentation details from API consumers and provide a simple "here-do-this-next" style pagination interface, instead.

For example, let's say you wanted to use the API to fetch all the localities (there are over 200,000 of them) in sets of five. That will require more than 41,000 API requests but that's your business. The API will respond with a `next_query` parameter, something like this:


```
{
    "places": [ ... ],
    "next_query": "method=whosonfirst.places.search&placetype=locality&per_page
    "total": 208214,
    "page": null,
    "pages": 41643,
    "per_page": 5,
    "cursor": "{CURSOR}",
    "stat": "ok"
}
```

There are a few things to note about the `next_query` property:

- ***It contains a URL-encoded query string with the parameters to pass to the API retrieve the next set of results for your query.***
- ***When it is empty (or `null`) that means there are no more results.***
- ***It does not contain any user-specific access tokens or API keys — you will need to add those yourself.***
- ***It does not contain any host or endpoint specific information — you will need to add that yourself.***
- ***You may want or need to decode the query string in order to append additional parameters (like authentication) and to handle how those parameters are sent along to the API. For example, whether the method is invoked using HTTP's `GET` or `POST` method or whether parameters should be `multipart/mime` encoded or not. And so on.***

This type of pagination is not ideal but strives to be a reasonable middle-ground that is not too onerous to implement

and easy to use.

Pagination and HTTP headers

Pagination properties are also returned as HTTP response headers. This is useful for any output format and necessary for output formats like plain old CSV `#formats-csv` or Who's On First's **meta** `#formats-meta` format. All of the pagination properties you've come to know and love in the examples above are also returned as HTTP response header prefixed by `X-api-pagination-`.

For example:

```
$> curl -s -v -X GET 'https://whosonfirst-api.mapzen.com/?method=whosonfirst.pl

< HTTP/1.1 200 OK
< Access-Control-Allow-Origin: *
< Content-Type: text/csv
< Date: Tue, 28 Feb 2017 21:13:37 GMT
< Status: 200 OK
< X-api-pagination-cursor:
< X-api-pagination-next-query: method=whosonfirst.places.search&q=poutine&extra
< X-api-pagination-page: 1
< X-api-pagination-pages: 13
```

```
< x-api-pagination-per-page: 1
< X-api-pagination-total: 13
< X-whosonfirst-csv-header: geom_bbox,wof_country,wof_id,wof_name,wof_parent_id
< Content-Length: 208
< Connection: keep-alive

geom_bbox,wof_country,wof_id,wof_name,wof_parent_id,wof_placetype,wof_repo
"-71.9399642944,46.0665283203,-71.9399642944,46.0665283203",CA,975139507,"Pouti

Libraries
```

Libraries

The Kitten with the Big Green Eyes

Music by DEL SHARBUTT · *Words by* AL VANN A. S. C. A. P. *and* FRANK STANTON

FAMOUS MUSIC CORPORATION · 1619 Broadway · New York City, N. Y.

As of this writing there are two general-purpose software libraries for accessing the Who's On First API, one written in **Python** `https://github.com/whosonfirst/py-mapzen-whosonfirst-api` and another in **Go** `https://github.com/whosonfirst/go-whosonfirst-api` . We also have PHP and Javascript libraries that we use internally but they are still full of Mapzen-isms which we'd like to clean up before releasing.

The Python and Go libraries don't have feature parity and probably never will. They are each well-suited for different tasks and, as with databases, we use their respective strengths and weaknesses to test the decisions we make modeling the actual Who's On First data. We use both of these libraries on a day-to-day basis. *Most* of the kinks should be worked out by now but the documentation could be improved in places, notably in the **Go** `https://github.com/whosonfirst/go-whosonfirst-api` package which we're using **as a way to test** `https://github.com/whosonfirst/go-whosonfirst-api#interfaces` how a strictly typed language can work with less formal and semi-structured API responses.

If you write your own library for the Who's On First API **we'd love to hear about it** `https://twitter.com/alloftheplaces` .

py-mapzen-whosonfirst-api

The Python library is available on GitHub: **https://github.com/whosonfirst/py-mapzen-whosonfirst-api** `https://github.com/whosonfirst/py-mapzen-whosonfirst-api` . Here's a simple example of how you might use it:

```
import mapzen.whosonfirst.api.client

api = mapzen.whosonfirst.api.client.Mapzen("mapzen-xxxxxx")
print api.execute_method("api.spec.formats", {})

# prints:
# {u'default_format': u'json', u'stat': u'ok', u'formats': [u'json', u'csv', u'
```

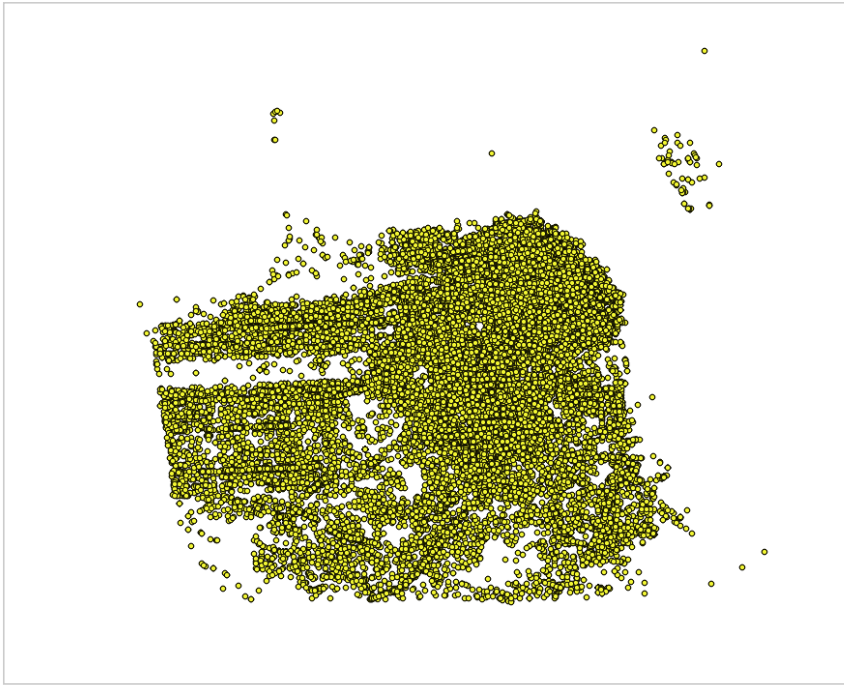
go-whosonfirst-api

The Go library is available on GitHub: **https://github.com/whosonfirst/go-whosonfirst-api** `https://github.com/whosonfirst/go-whosonfirst-api` . Here's a simple example of how you might use it:

```
import (
    "github.com/whosonfirst/go-whosonfirst-api/client"
    "github.com/whosonfirst/go-whosonfirst-api/endpoint"
    "os"
    )

func main() {

    api_key := "mapzen-xxxxxx"
```

```
    api_endpoint, _ := endpoint.NewMapzenAPIEndpoint(api_key)
    api_client, _ := client.NewHTTPClient(api_endpoint)

    method := "whosonfirst.places.search"

    args := api_client.DefaultArgs()
    args.Set("query", "poutine")
    args.Set("placetype", "venue")

    rsp, _ := api_client.ExecuteMethod(method, args)
    os.Stdout.Write(rsp.Raw())


    # prints (truncated for brevity)
    # {"places":[{"wof:id":152777717,"wof:parent_id":"85874363","wof:name":"Pou
}
```

There is also a command line tool called `wof-api` for performing simple and batch operations with the API. For example, using the tool it's possible to **fetch all 63,387 venues in San Francisco** `https://github.com/whosonfirst/go-whosonfirst-api#example` as a single GeoJSON FeatureCollection. There are lots of ways to do this but the nice thing about the `wof-api` tool is that depending on your network connection this can be accomplished in as little as 5 minutes.



All those dots in the image above were produced by typing

```
wof-api -param
method=whosonfirst.places.search -param
locality_id=85922583 -param api_key=mapzen-
xxxxxx -param per_page=500 -param
placetype=venue -paginated -geojson -output
venues.geojson -timings -async
```

Note the use of the `-paginated` flag which will instruct the tool to handle all the pagination nonsense described above for you. It's kind of like having your very own **Bundler** `https://mapzen.com/blog/bundler/` on the **command line** `https://en.wikipedia.org/wiki/In_the_Beginning..._Was_the_Command_Line` .

Inspirational conclusion

We hope you make awesome things with the API. Enjoy!

Image credits:

- ***Model Of A Walled Entrance With Gate (Italy), ca. 1710; carved and painted pine, wrought iron gate and window grill; Gift of Mrs. James O. Green and Eleanor and Sarah Hewitt; 1921-17-16-a/d*** https://collection.cooperhewitt.org/objects/18214745/

- ***Label (USA), 1980–85; jacquard woven; 1985-39-60*** https://collection.cooperhewitt.org/objects/18805295/

- ***Sheet Music, The Kitten with the Big Green Eyes; 1949-106-3*** https://collection.cooperhewitt.org/objects/18612725/

- ***Panel (England), mid-19th century; cotton; Gift of Ruth Van Norman; 1955-119-1*** https://collection.cooperhewitt.org/objects/18402495/

2017-04-04

things I have written about elsewhere #20170428

The Who's On First API Explorer

The Who's On First API Explorer

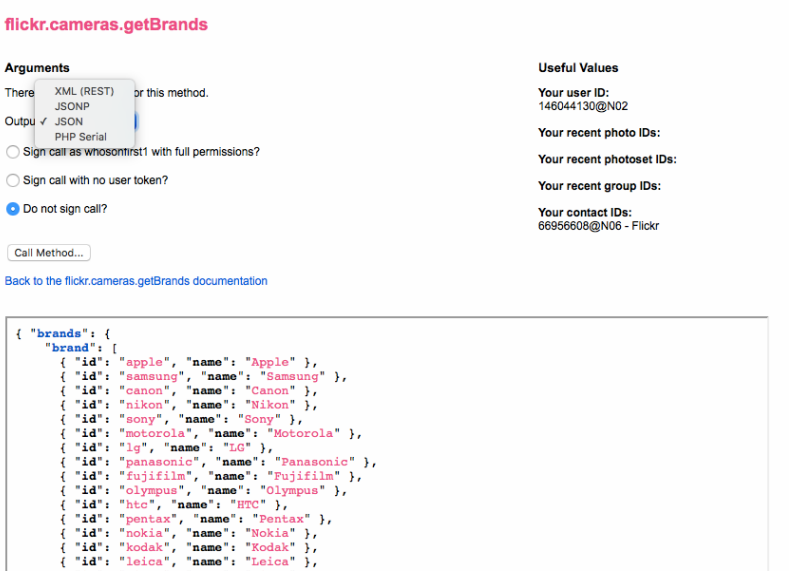
*This was **originally published on the Mapzen weblog*** `https://mapzen.com/blog/whosonfirst-api-explorer/` *, in April 2017.*

A few weeks ago **Lou built an Electron application** `https://mapzen.com/blog/tangram-work-developer-commentary` . For April Fools' Day he made a version of **Tangram Play** `https://mapzen.com/blog/introducing-tangram-work` in the style of a 90's era Windows application and published it as a living breathing desktop application rather than **the web application** `https://mapzen.com/tangram/play/` we've come to know and love. In addition he was able to publish MacOS and Linux native applications, all from the same code base. The same code base that is ultimately just HTML and

CSS and Javascript (aka "a web application"). An **increasing number of desktop applications** https://electron.atom.io/apps are being developed this way and it's what makes Electron pretty exciting.

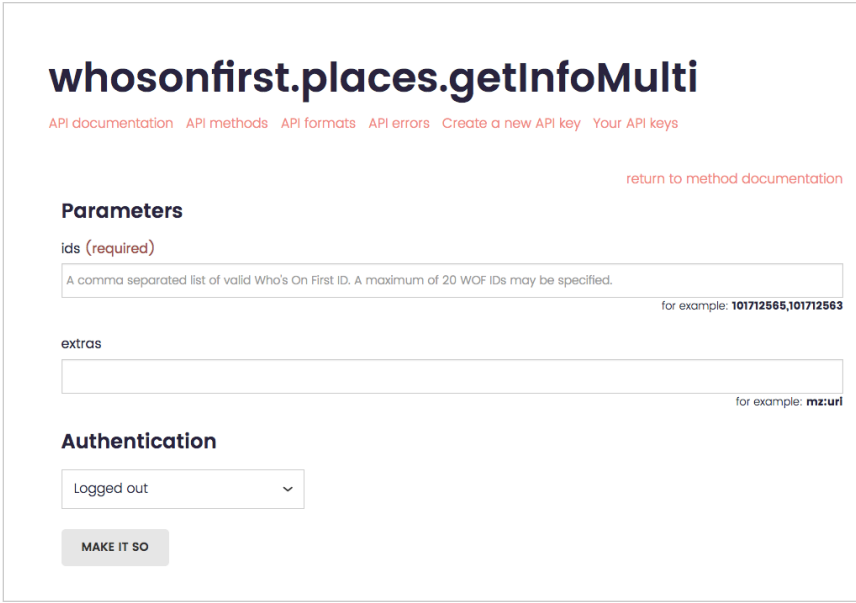
Shortly after Lou's blog post we pushed the **Who's On First API** https://mapzen.com/blog/whosonfirst-api out the door. One of the features of the API is that it exposes a suite of `api.spec.*` methods **that allow you to introspect the API itself** https://mapzen.com/documentation/wof/methods/#apispec . You can fetch all of the **methods** https://mapzen.com/documentation/wof/methods/#api.spec.methods (and their parameters and errors), all of the **common errors** https://mapzen.com/documentation/wof/methods/#api.spec.errors and all of the supported **output formats** https://mapzen.com/documentation/wof/methods/#api.spec.formats . These methods owe a debt of inspiration to the nice people at **Linode** http://linode.com who started doing something similar **way back in 2012** http://blog.linode.com/2012/04/04/api_spec/ and from which the `api.spec` naming convention is derived.

The Who's On First API has always enjoyed a internal web-based "explorer" tool. API explorers are pretty common these days but if you're not sure they are **have a look at the Flickr API Explorer** https://www.flickr.com/services/api/ which is pretty much the ur-Explorer that started them all. Go ahead, we'll wait...



*Also, Cal Henderson's 2007 talk "**Flickr and APIs: A Love Story**"* `http://www.iamcal.com/talks/open_hack_day.pps` *which is a 22MB PowerPoint presentation because... it was 2007, I guess. Anyway, it it a good talk.*

We use our own API Explorer extensively for debugging and generally poking around. It is often the fastest way to answer a question about the data. I find it difficult to imagine developing or working with APIs *without* an API explorer now and that makes the internal-iness of the WOF API Explorer less than awesome. For a variety of reasons it won't be made public any time soon which is sort of like pouring salt in your own wound.



By now, some of you might have figured out where this is going: If the WOF API has a full suite of methods for inspecting the API and Electron makes it easy to develop custom applications in HTML and Javascript then why not build... **a standalone Who's On First API Explorer** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer`. So we did!

After I showed an early version to a friend he asked "*Why an Electron application?*" for an API explorer. Aside from the reasons outlined above I offered the following rationale:

1. *Curiousity. Does the Who's On First API lend itself to an Electron application? (It does!)*

2. *If you set aside some boring details about code-signing and paying Apple or Microsoft for the privilege, there is the possibility of generating native platform specific binaries for Electron applications. That means never having to say "Just install Node.js..." to someone. You can replace* ***Node.js*** `https://nodejs.org/` *with any number of technologies and the dynamic is the same. Tools that don't require a laundry list of extra steps to get started with are important. Not everyone is interested in the technology details because they are busy being awesome in other endeavours.*

3. *It was (is) a good way to put the* `api.spec` *methods through their paces. We've been able to identify a bunch of places where the API doesn't return enough information to usefully build an explorer-style interface. Many have been fixed already and the rest (notably information about output formats) are slated to be improved shortly.*

4. *It allows us offer a public version of the API explorer.*

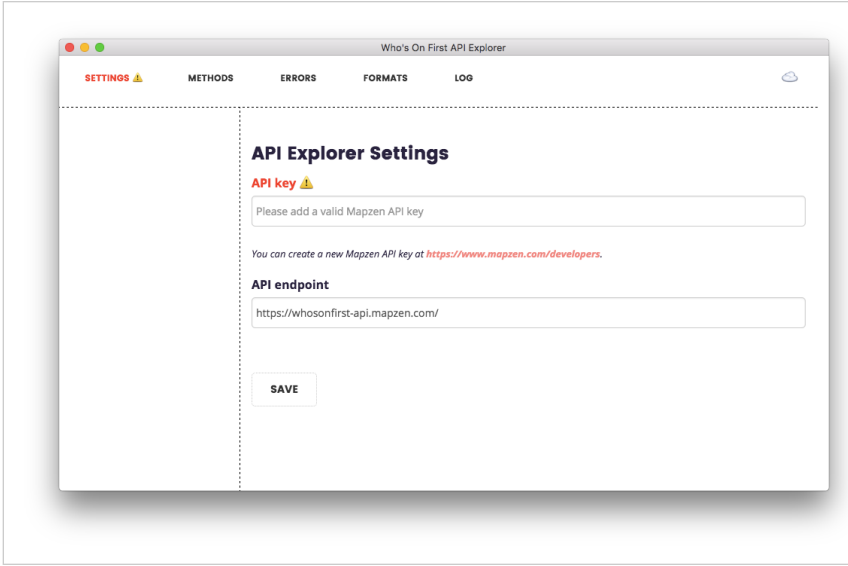
5. *It is a good way to figure out which out* ***the Who's On First Javascript libraries*** `https://github.com/whosonfirst/js-mapzen-whosonfirst` *should be rejiggered to hold hands with "modern Javascript" and Node.js – things like the* ***mapzen.whosonfirst.api.js*** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer/blob/master/mapzen.whosonfirst.api.js` *library which we'd like to bless as officially supported alongside the* ***Go*** `https://github.com/whosonfirst/go-whosonfirst-api` *and* ***Python*** `https://github.com/whosonfirst/py-mapzen-whosonfirst-api` *libraries.*

6. *It works offline because initial calls to the* `api.spec.*` *methods are cached locally.*

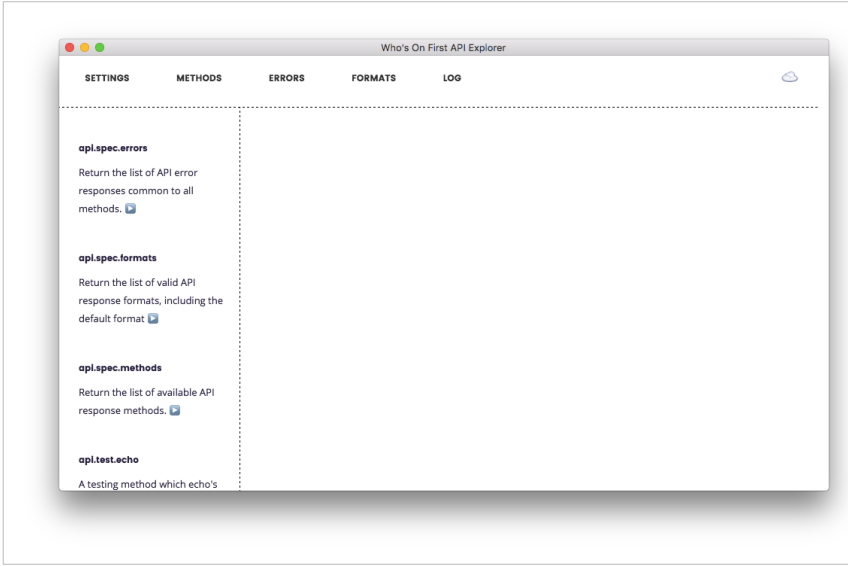
I like to think the WOF API Explorer is another illustration of the idea that **"Mapzen should always be Consumer Zero (of Mapzen services)"** `https://mapzen.com/blog/iamhere` . The WOF API Explorer is just one of many possible **targeted and focused applications** `https://mapzen.com/blog/bundler` that we might build with tools like Electron and **our own services** `https://www.mapzen.com/documentation` .

The **WOF API Explorer is available for anyone to try** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer` . As of this writing, and despite what I said above, you will still need to "*Just install Node.js*" to get it to work. Hopefully the time when we dig in to the details of building and signing native applications will come soon. Until then there is **a little bit (possibly a lot, depending on your skill and comfort level) of preamble** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer#install` you'll need to do to get the WOF API Explorer up and running.

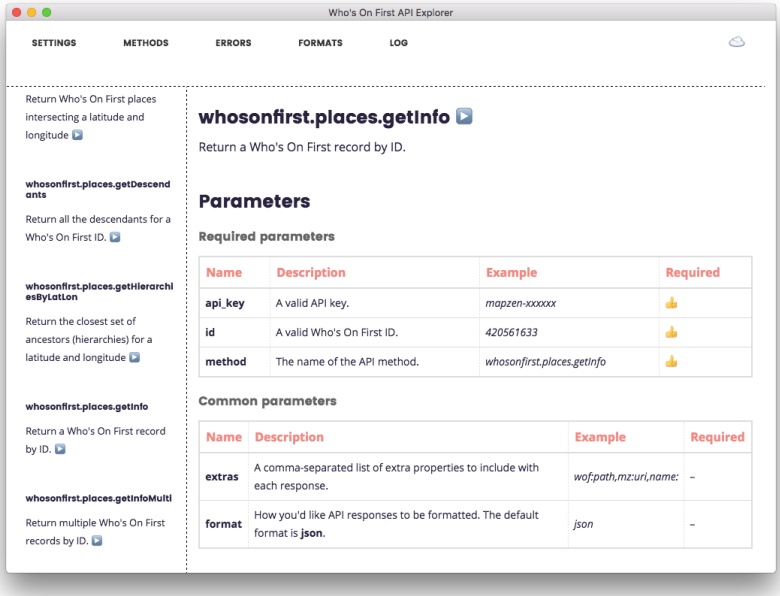
In the interim, here are a bunch of screenshots to hopefully make that extra work seem worth doing:



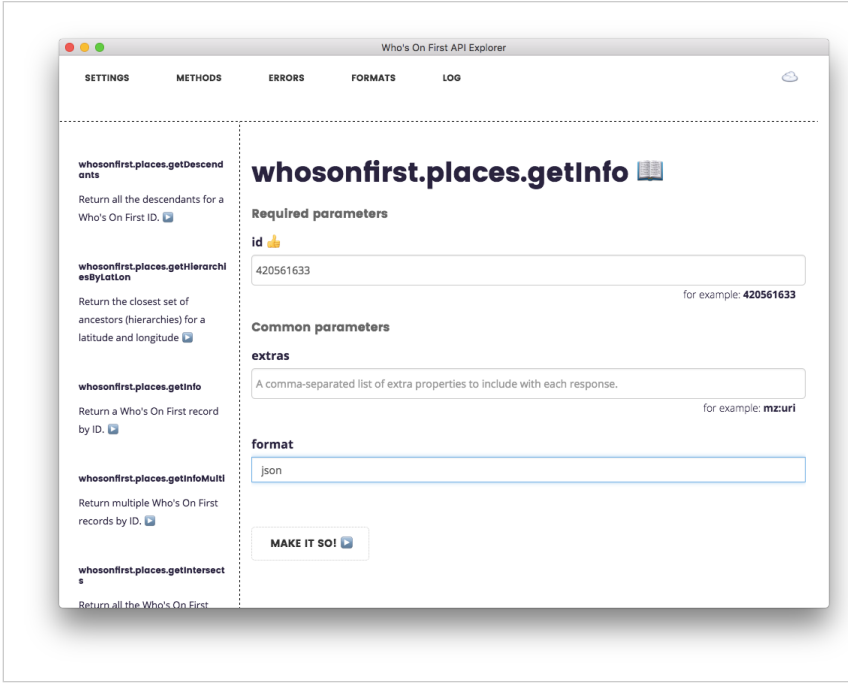
Here's what happens when you start the API Explorer for the first time: ***You are asked for a Mapzen API key!*** `https://mapzen.com/developers`



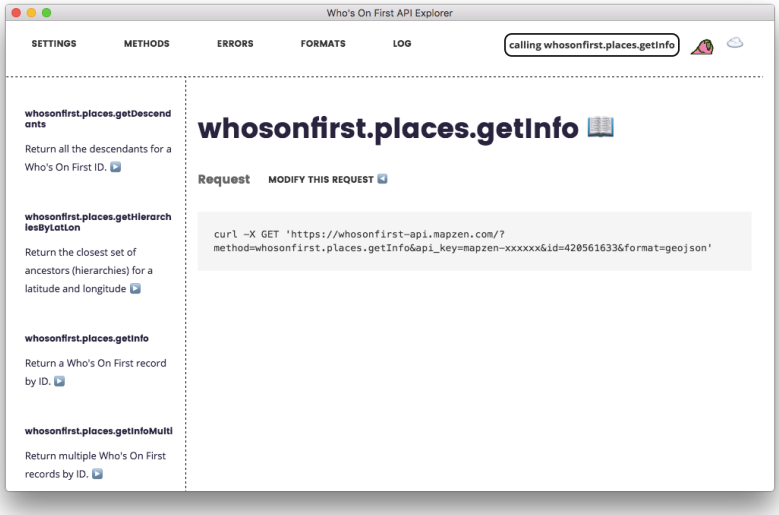
List views appear in the left-hand sidebar. Here's a list of all the API methods. API methods, errors and formats are fetched from the API when the Explorer starts up and there is a handy "refresh" button at the bottom of each list view update things while the application is running.



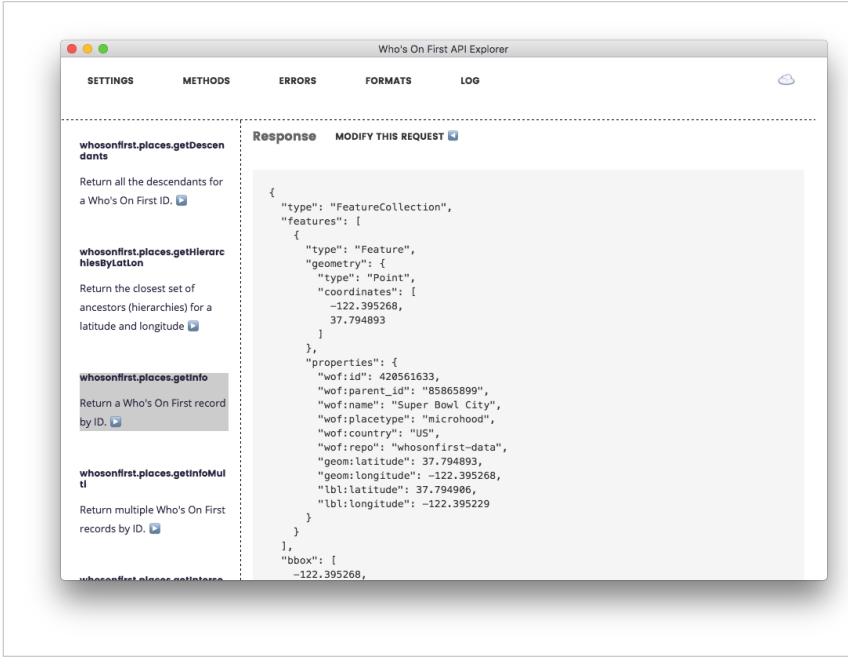
Here's an example of the documentation for the ***whosonfirst.places.getInfo*** `https://mapzen.com/documentation/wof/methods/#whosonfirst.places.getInfo` *API method, which isn't that different from what you'd see on the* ***Mapzen website*** `https://mapzen.com/documentation/wof/` .



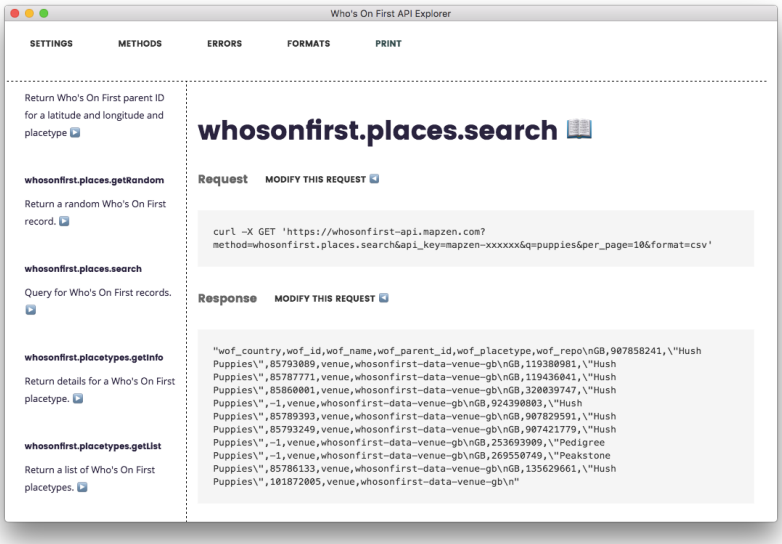
From any documentation page you can click through to a corresponding "explore" panel for that method. You can update the parameters that will be sent to the API and then click the handy `MAKE IT SO!` *button to start the API request.*



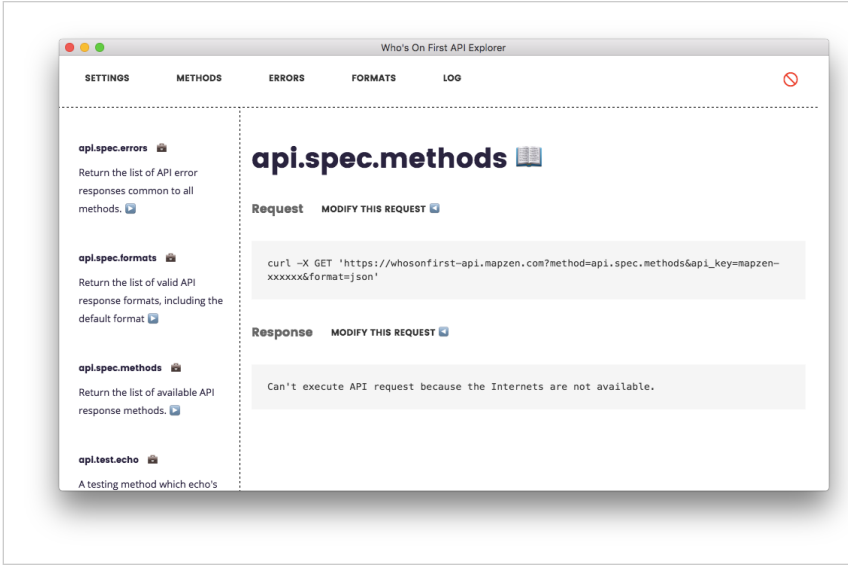
:partyparrot: `http://aaronland.info/weblog/2017/04/28/things/images/party-parrot.gif` *will keep you company while your API request is being processed.*



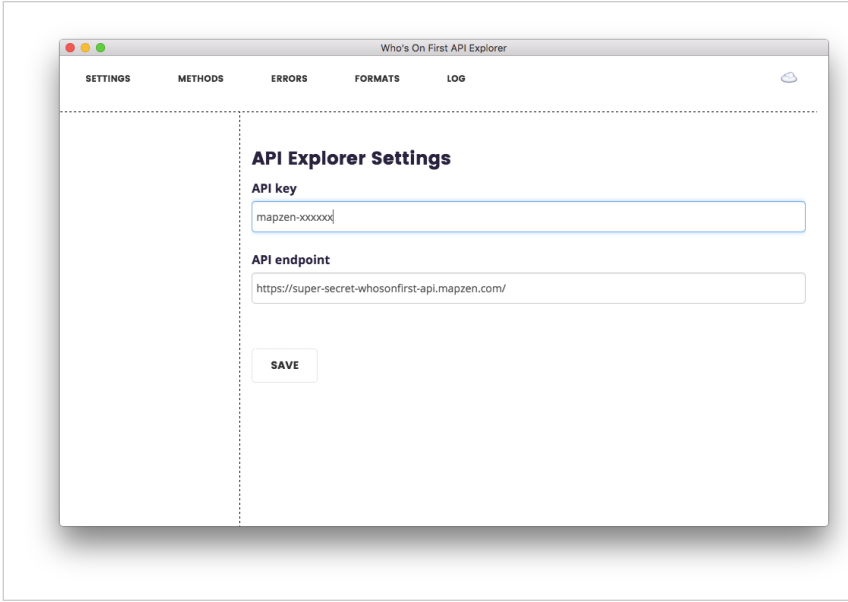
Oh look, here's a new part of the Who's On First API: ***GeoJSON*** `https://mapzen.com/documentation/wof/formats/#geojson` *as an output format.*



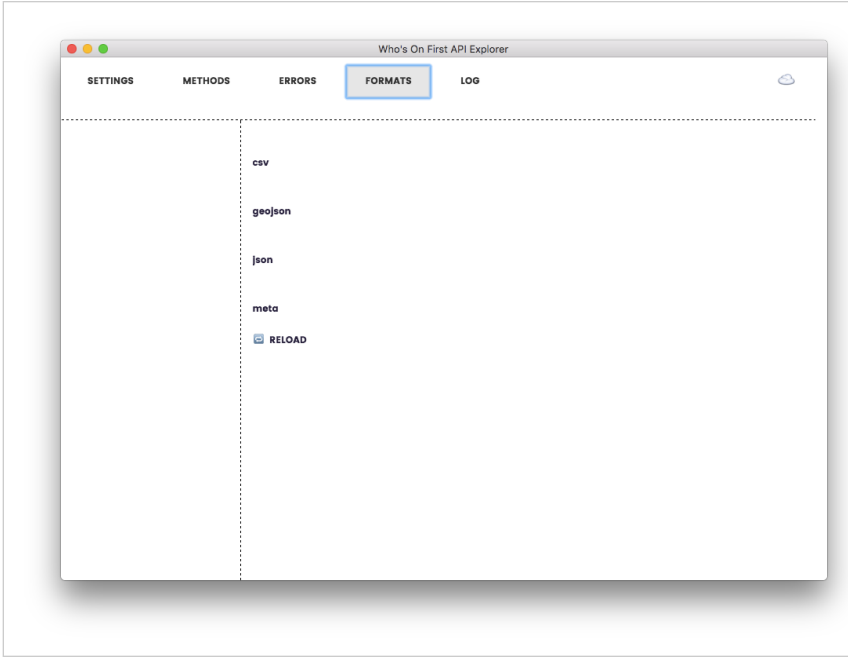
The API Explorer can also render non-JSON output formats. There is also ***an open ticket to enable downloading of results*** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer/issues/21` *from the API Explorer.*



Importantly the API Explorer will continue to work offline. See the way the ☁️ in the top-left corner became a 🚫 - that means the API Explorer can't find the Internets. You won't be able to send API requests out across the network but you can still read the documentation and construct API requests to review.

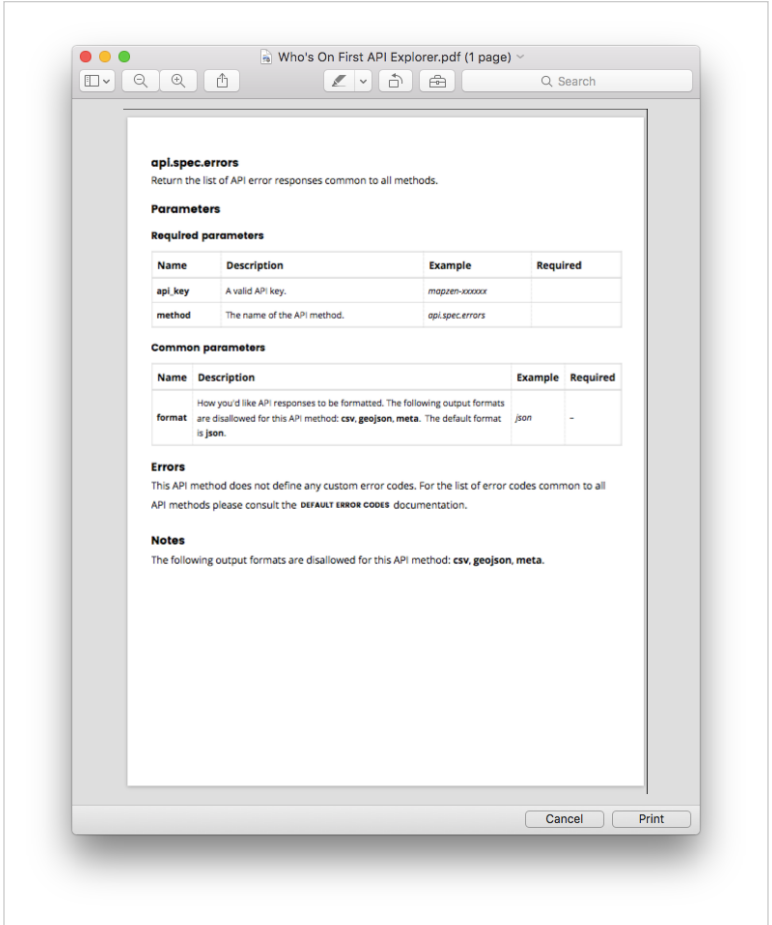


You can also change the API endpoint for making requests and by extension information about the API itself. via the `api.spec` *methods. This probably isn't that useful to anyone else right now but it's very helpful for us when testing new API methods on our dev servers. Eventually* ***we'd like to support "named" and "toggle-able" configurations*** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer/issues/31` *in the API Explorer to make it easy to switch from one API endpoint to another.*



Most of the work, in both the API and the Explorer, has been around API methods. There is still some more work to do for the other things...

There's an ***an experimental and currently disabled print feature*** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer/issues/10` *to generate print-friendly version of both API documentation and results.*



Unfortunately, I can't figure out why print versions are truncated after one "page" so we'll turned off that functionality for now. Also apparently Emoji (specifically the 👍 signifying a required parameter) are silently dropped in the print output because… computers?

Like the API itself, the API Explorer still tilts towards beta as of this writing. There are a number of **outstanding issues** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer/issues`, most of them minor and nothing that should prevent you from using the API Explorer from... exploring the **Who's On First API** `https://mapzen.com/documentation/wof`.

The source code is available from **https://github.com/whosonfirst/electron-whosonfirst-api-explorer** `https://github.com/whosonfirst/electron-whosonfirst-api-explorer` and we would welcome your feedback and suggestions.

Image credits:

- ***Drawing, To Explore by that Cheerless Illumination for Leaks, ca. 1904; Frank Walter Taylor ; USA; point of brush and black, gray wash, white goache, black crayon on bristol board; Sheet: 44.4 x 30.6 cm (17 x 12 1/16 in.); Gift of F. Elizabeth De Voy; 1957-58-15*** `https://collection.cooperhewitt.org/objects/18419753/`

2017-04-28

things I have written about elsewhere #20171017

maîtres chez nous

maîtres chez nous

*This was **originally published on the Mapzen weblog*** `https://mapzen.com/blog/whosonfirst-nacis-2017/` *, in October 2017.*

Hello.

It is a pleasure to be back with you at **NACIS** `http://nacis.org/` . The last time I was here **was in 2011** `http://www.aaronland.info/weblog/2011/10/14/pixelspace/#nacis` . It is also a treat for me that the conference is being

held in **Montreal** `https://places.mapzen.com/id/101736545/` because this is where I grew up.

Thank you for inviting me to speak.

This is a talk about gazetteers and in particular the gazetteer that I work on, called **Who's On First** `https://whosonfirst.mapzen.com` . Who's On First has set itself the ambitious goal of global coverage from continents all the way down to venues.

For anyone not familiar with the term a "gazetteer" is basically just a great big phone book, but a phone book of places rather than people.

Gazetteers have existed for as long as we've been able to conceptualize the idea of place. http://www.iupress.indiana.edu/product_info.php?products_id=808056 A gazetteer is so deceptively simple as to seem mundane and beyond effort but so complex, the moment you scratch the surface, as to overwhelm even the best intentions.

This is not a talk about the mechanics of Who's On First. In the two and half years since the project began **something like forty to fifty thousand words have been written** https://whosonfirst.mapzen.com/blog about the architectural theory and the engineering choices and all the challenges we've faced along the way.

This is not a talk about how we work on a gazetteer like Who's On First but rather why we work on it, at all.

If not how but instead why, we still need to start with what. What is the shape of the Who's On First gazetteer?

- ***To begin, Who's On First is an openly licensed*** `https://whosonfirst.mapzen.com/docs/licenses/` ***dataset. At its most restrictive, data is published under a Creative Commons By-Attribution license. Whenever possible, and this is true of our own day-to-day work, data is published under a Creative Commons Zero public domain license.***

- ***Every record in Who's On First has a stable permanent and unique numeric identifier*** `https://whosonfirst.mapzen.com/data/principles/` ***. There are no semantics encoded in the IDs.***

- ***At rest, each record is stored as a plain-text GeoJSON file. Our goal is to ensure that Who's On First embodies the principals of portability, durability and longevity. This led us to adopt plain-vanilla text files as the base unit of delivery.***

- ***Files are stored in a nested hierarchy of directories derived from their IDs.***

- ***There are a common set of properties*** `https://whosonfirst.mapzen.com/docs/properties/` ***applied to all records which may be supplemented by an arbitrary number of additional properties specific to that place.***

- ***There are a finite number of place types*** `https://github.com/whosonfirst/whosonfirst-placetypes#here-is-a-pretty-picture` ***in Who's On First and all records share a common set of***

ancestors. As with properties, any given record may have as complex a hierarchy as the circumstances demand `https://whosonfirst.mapzen.com/docs/hierarchies/` *but there is a shared baseline hierarchy across the entire dataset.*

- *Individual records may have multiple geometries or multiple hierarchies and sometimes both.*

- *Records may be updated or superseded, cessated or even deprecated. Once a record is created though it can never be removed or replaced.*

- *Lastly and most importantly Who's On First is meant, by design, to accommodate all of the places.*

Who's On First is not a carefully selected list of important or relevant places. It is not meant to act as the threshold by which places are measured.

Who's On First, instead, is meant to provide the raw material with which a multiplicity of thresholds might be created.

From Who's On First's perspective **the point is not that one place is more important or relevant than any other** `https://mapzen.com/blog/who-s-on-first/` . The point is not that a place may or may not exist anymore or that its legitimacy as a place is disputed. The point is, critically, that people *believe* them to exist or to have existed.

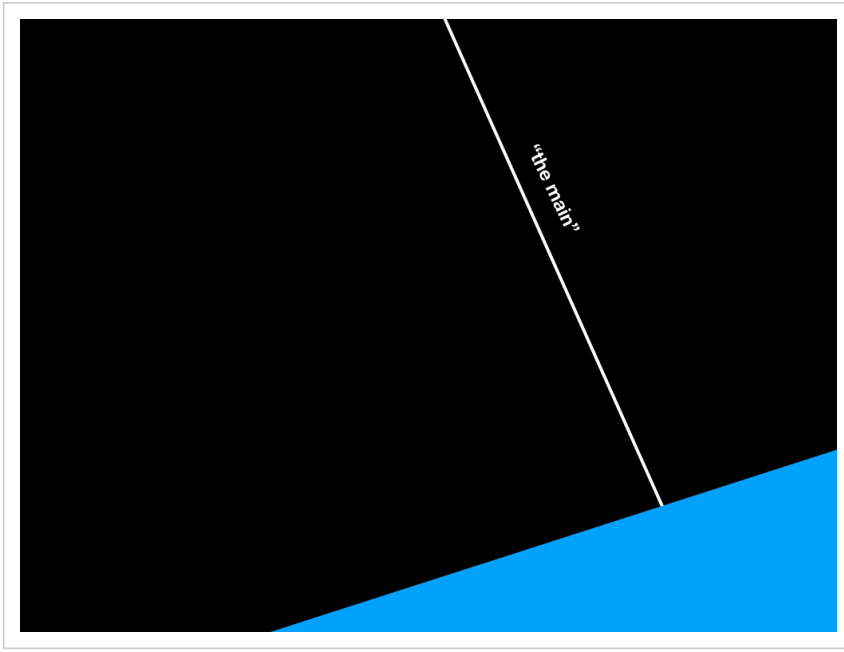
This is why I often refer to Who's On First as **"a gazetteer of consensual hallucinations"** `https://mapzen.com/blog/mapping-with-bias/` .

To explain why I'm going to start with this guy.

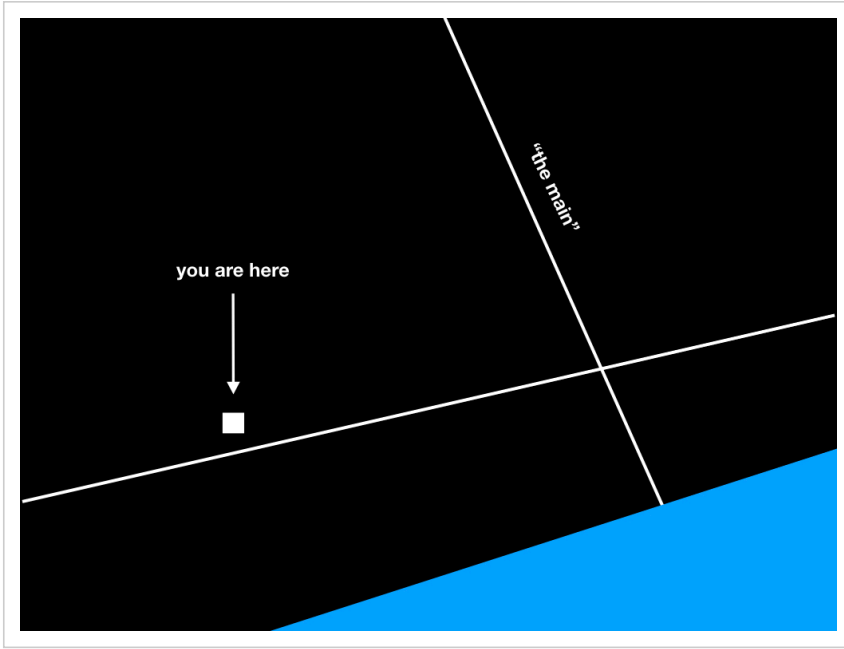
This is **Leonard Cohen** http://www.pbs.org/video/montreal-ladies-and-gentlemen-mr-leonard-cohen/ , the poet and musician, who **died in November** https://www.youtube.com/watch?v=jP7pN_3_JQI of last year. Cohen was born and raised in Montreal and **this mural** http://montrealgazette.com/news/local-news/leonard-cohen-looms-large-at-montreals-mural-

international-public-art-festival was **painted** http://www.kevinledo.com/ on St. Laurent Boulevard, just **down the street** https://www.nytimes.com/2017/02/10/travel/leonard-cohen-musician-montreal-canada.html from where he kept a house in the city.

St. Laurent Boulevard is often referred to by locals simply as "The Main".



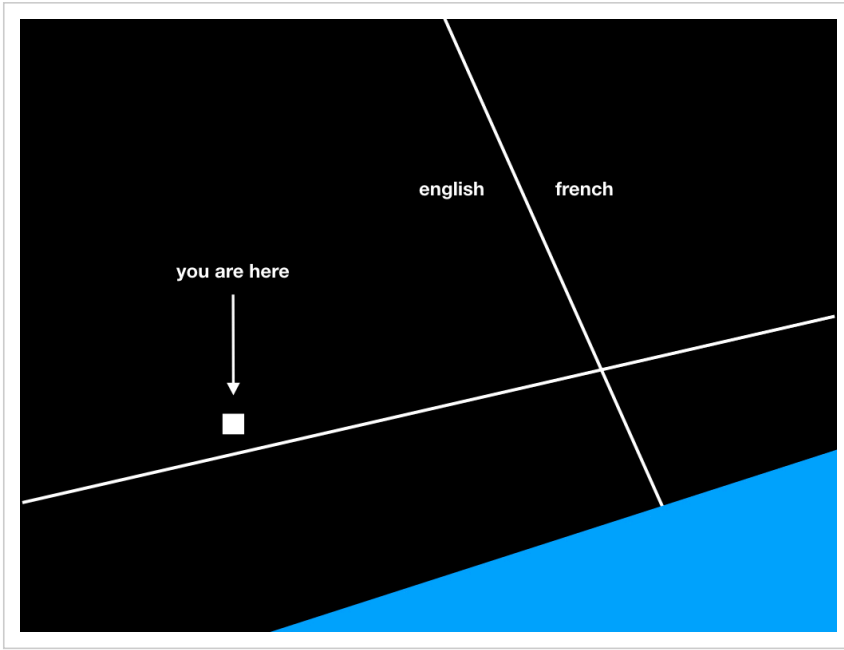
The Main is so-called because it is one the city's oldest streets running along a North-South axis, starting in Old Montreal at the bank of the **St. Lawrence River** `https://places.mapzen.com/id/404529181/` .



Like a lot of cities in the world, Montreal's North-South axis has no bearing in reality.

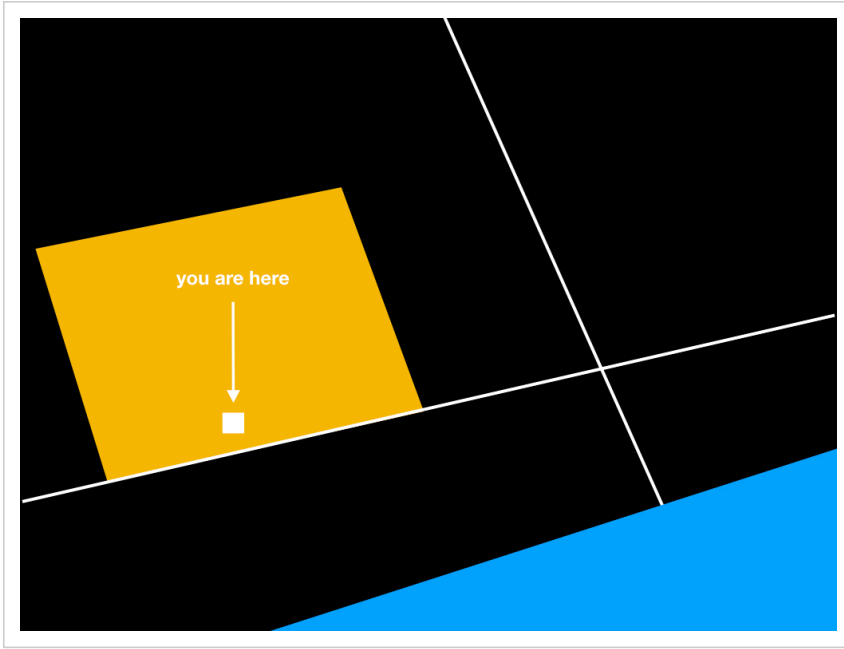
If you're being precise St. Laurent Boulevard runs along a North-North-West to South-South-East axis.

What's important about The Main is less its directionality than the fact that it acts as the border between the Eastern and Western halves of the city.



It is only in recent years that The Main has not served as the dividing line between the city's English and French speaking populations.

This was famously described by the author **Hugh MacLennan** `http://www.hipmuseum.com/hugh.html` in the larger Canadian context as "two solitudes" living side by side **with little or no common ground between them** `http://archive.gg.ca/media/doc.asp?lang=e&DocID=4574` .



It is also not really possible to talk about that divide without talking about it as the border between the haves and the have-nots.

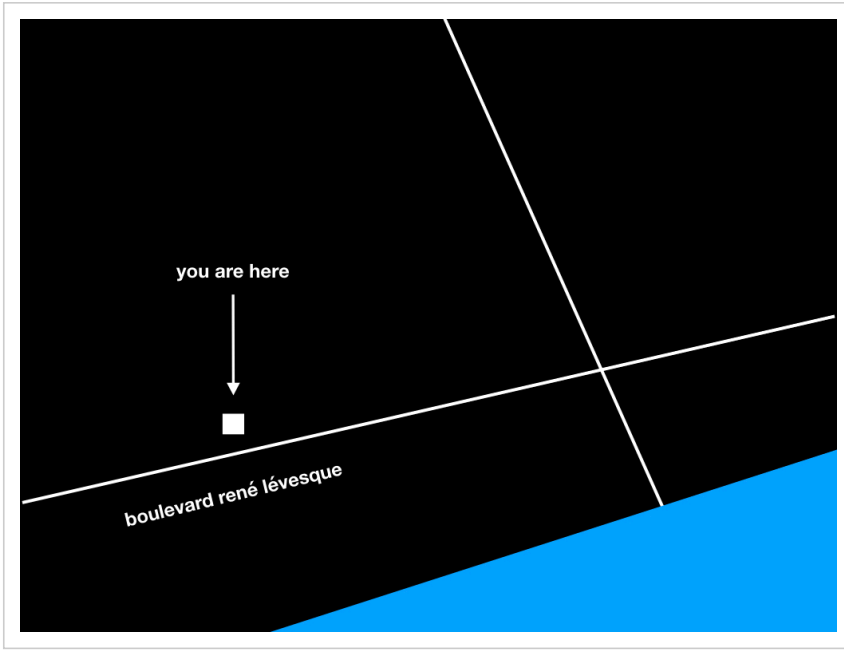
This conference is being held on the Southern edge of what was once called **The Golden Square Mile** `https://places.mapzen.com/id/1108955947/` . This part of town is where what we now call "the 1%" lived.

For a time this was the seat of all the wealth and power in Canada. It was very very English at a time when the contempt and

disdain between the English and the French was barely contained and rarely disguised.

This created a power dynamic that rippled down through every aspect of society, in a province that has always been 90% Francophone. It was a power dynamic that was not only allowed to **fester** https://www.urbandictionary.com/define.php?term=T%C3%AAte%20carr%C3%A9e but actively promoted.

This continued to varying degrees until the 1950s and what is known as **the Quiet Revolution** http://archives.radio-canada.ca/politique/provincial_territorial/dossiers/1732/ when the French Canadian population demanded, not for the first time, to be **"maîtres chez nous"** https://en.wikipedia.org/wiki/Quiet_Revolution or "masters in our own home."



You may have noticed that the conference hotel sits on a street named after a guy called **René Lévesque** `https://en.wikipedia.org/wiki/Ren%C3%A9_L%C3%A9vesque` .

It is difficult to overstate Lévesque's role during **the Quiet Revolution** `http://www.banq.qc.ca/collections/collections_patrimoniales/bibliographies/revolution_tranquille.html` and in shaping contemporary life and the social contract in Québec.

He would be elected Premier in 1976 as the head of the separatist Parti Québécois (PQ) and would lead **an unsuccessful bid to secede from Canada in 1980** https://www.youtube.com/watch?v=lru4grpq3Rc .

Note: During the talk I incorrectedly stated that "Among other things he gave Québec a Charter of Human Rights and Freedoms almost a decade before Canada ***enacted its own*** http://laws-lois.justice.gc.ca/eng/Const/page-15.html *and language laws which are still being debated, to this day."*

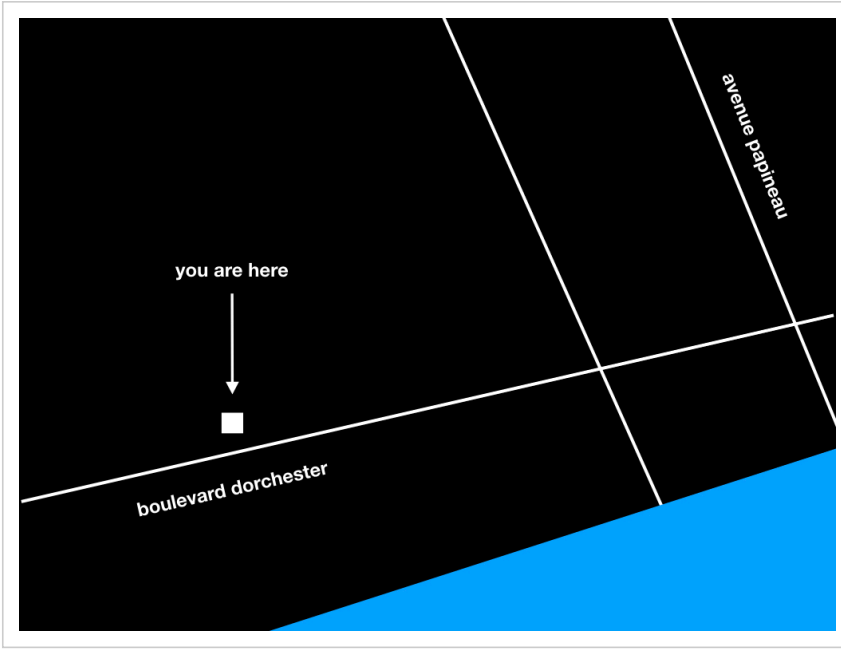
In fact, The ***Québec Charter of Human Rights and Freedoms*** https://en.wikipedia.org/wiki/Quebec_Charter_of_Human_Rights_and_Freedoms *was passed by the provincial National Assembly the year before the PQ were elected. That is my bad so I will point out, instead, that Lévesque both championed and oversaw the final* ***nationalization of hydroelectric power in Québec*** http://larevolutiontranquille.ca/en/the-nationalization-of-electricity.php *. While not necessarily as inspiring as a Charter of Human Rights and Freedoms it would have an equal (probably greater) impact on the province.*

By the time he died in 1987 few would deny Lévesque, whatever they thought of his politics, his place as **the father of**

modern Québec https://www.youtube.com/watch?v=BhbSp4cJZVY .

So you would be forgiven for wondering why we have chosen to honour such a celebrated figure with such an unremarkable boulevard. The next time you go outside take a moment to marvel at its mediocrity.

The answer lies in what Boulevard René Lévesque used to be called.



Boulevard René Lévesque used to be called Dorchester Boulevard after the English Lord Dorchester, who was formerly known as Lord Durham.

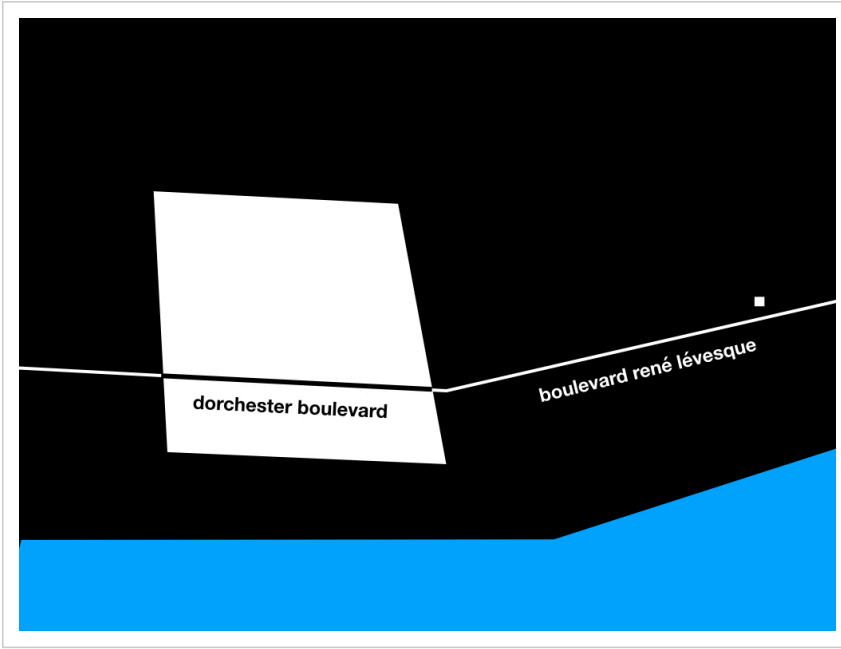
Lord Durham was sent over to Canada in 1838 to report on **the Patriot Rebellions** `https://fr.wikipedia.org/wiki/R%C3%A9bellion_des_Patriotes` of the previous year and more generally to determine why the Crown's French Canadian subjects were getting restless.

Fun fact: The rebellions were led by **Louis-Joseph Papineau** https://web.archive.org/web/20170612175735/https://www.pc.gc.ca/en/lhn-nhs/qc/louisjosephpapineau who was also honoured with an equally drab and uninspiring avenue about 40 minutes walk from here.

Lord Durham’s report https://en.wikipedia.org/wiki/Report_on_the_Affairs_of_British_North_America might have faded into historical obscurity had its author not offered the opinion that the French Canadians, who predated the first English settlers by a hundred years or more, were "un peuple sans histoire" or "a people without history."

It is not for nothing that all the license plates in this province now read **"je me souviens"** `https://www.youtube.com/watch?v=2FZXQZJxZRk` or "I remember".

Nor is it surprising that Dorchester Boulevard was purposely renamed to honour René Lévesque.

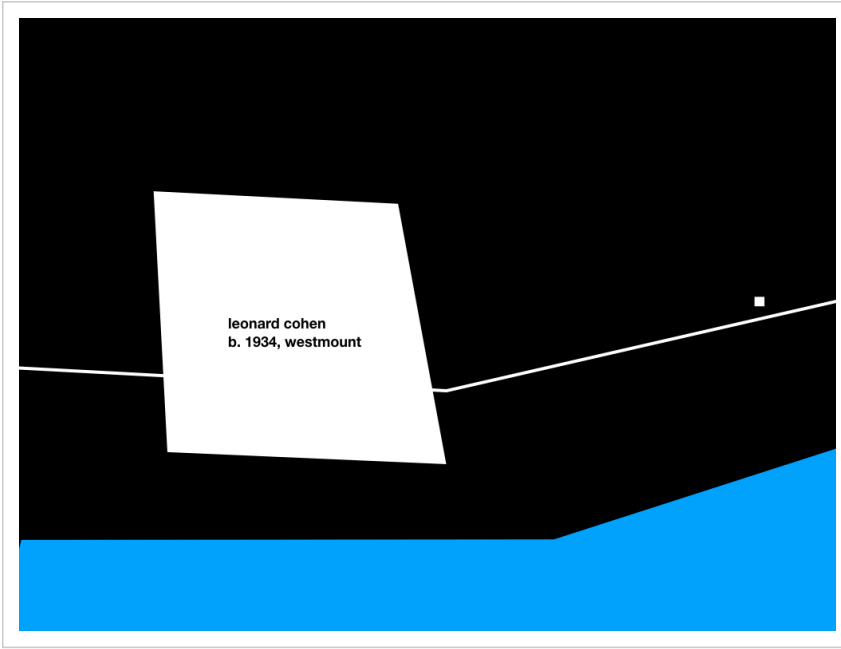


You know who else remembers, though? The city of **Westmount** `https://places.mapzen.com/id/101736451/` remembers.

Westmount is **an autonomous municipality** `https://westmount.org/en/the-city/history/` on the island of Montreal and home to the remaining money and power that didn't immediately decamp to **Toronto** `https://places.mapzen.com/id/101735835` following

the **election of René Lévesque** http://collections.musee-mccord.qc.ca/en/collection/artifacts/P090-A_50-1004/ .

Westmount is a sort of leftover bronze square mile of wealth and privilege and in 1987 it **refused to rename Dorchester Boulevard** https://www.westmount.org/wp-content/publications/A_View_Of_Their_Own_the_Story_of_Westmount/files/assets/basic-html/page151.html as the rest of the city had, and so it remains today.



Fun fact: **Leonard Cohen was born in Westmount** `http://www.leonardcohenfiles.com/montreal.html` , in 1934.

So, we're back to this guy.

By the time Leonard Cohen died last year in Los Angeles **he had lived there for decades** `http://www.cbc.ca/news/entertainment/leonard-cohen-la-home-1.3848347` . This pesky fact has never prevented anyone in Montreal, English or French, from claiming him **as one of their own** `https://www.newyorker.com/culture/culture-desk/leonard-cohens-montreal` .

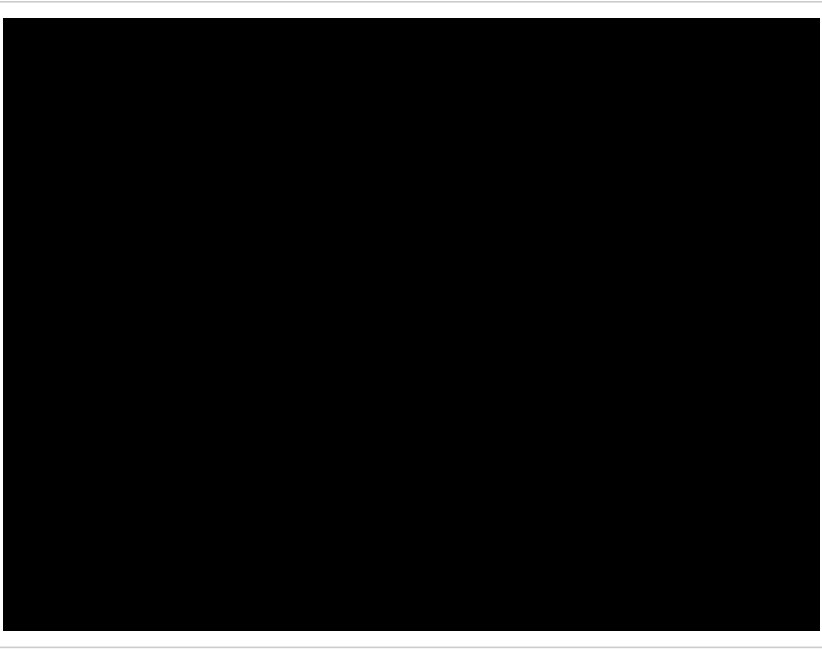
In the minds of many people in this city Leonard Cohen is before all things **a Montrealer** https://www.youtube.com/watch?v=ibUKEwds4Ng .

It is in all seriousness then that I would like to propose that the city of Montreal rename St. Laurent Boulevard after Leonard Cohen.

The simple fact is that Montreal suffers from a surplus of streets and plazas and landmarks devoted to the religious and linguistic power struggles and score-settling of the past.

We are overdue something that reflects the present. **We are overdue a celebration** `https://www.youtube.com/watch?`

v=2FpwjQLZTTs .



I have told you these stories for two reasons:

The first is that all of these things really happened and they are the stories that inform the city you'll be visiting during this conference.

The second is to use the telling – and in particular the telling of a duality, of ”two solitudes" and "two founding nations" – to highlight an uncomfortable fact about Canadian history, French and English both.

Simply put, we have been blind to the fact that there has always been another story. We have been blind to the fact that the First Nations were already here living on these lands long before the European settlers arrived.

It is important to recognize that we have not been passive in our blindness but brutally deliberate. First out of malice and then later out of negligence and more recently out of shame.

We have closed our eyes to the fact that almost from the beginning we have systematically worked to erase the First Nations, hoping only to open them when the deed is done and there is nothing left to remember but the two solitudes.

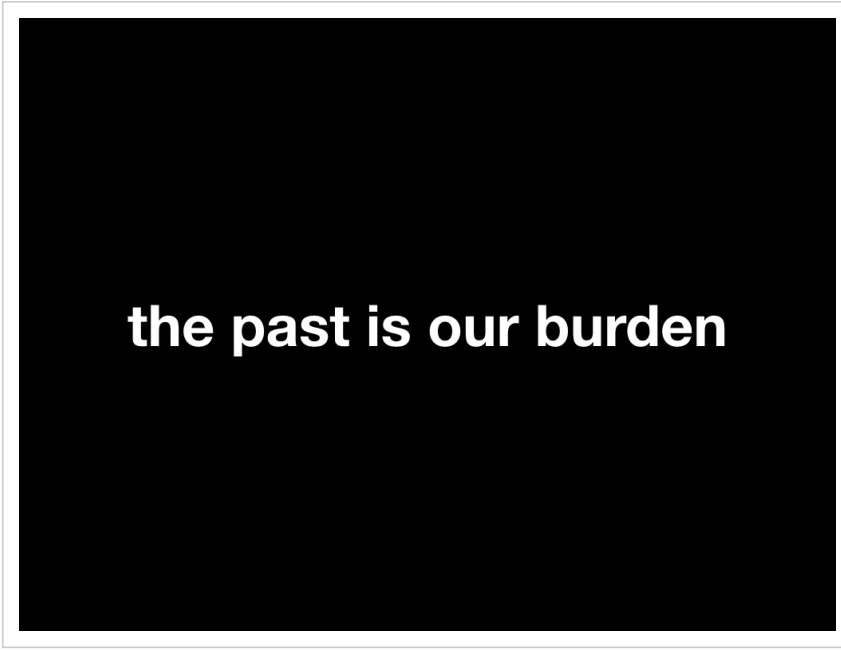
In 2017 there is at least a nascent awareness, something approaching acknowledgement, among Canadians that **these things happened and that we did them** `https://gadebate.un.org/en/72/canada` .

We are a long way from making amends and **there is not time enough to catalog all the ways we have done wrong** `http://www.trc.ca/websites/trcinstitution/index.php?p=3` by the First Nations but I want to touch on one story in particular.

As late as the 1950s and the 1960s it was, as part of official government policy to assimilate the First Nations, our practice **to forcibly remove First Nations children from their families** https://www.nytimes.com/2017/10/06/world/canada/indigenous-forced-adoption-sixties-scoop.html .

These children were placed in **foster homes or residential schools** http://www.aadnc-aandc.gc.ca/eng/1100100015576/1100100015577 where in addition to widespread physical and sexual abuse they were actively prevented from learning or practicing or remembering their language, their culture, their histories and in many cases their own birth names.

If you believe that place is history or that language is culture or that naming things is an important form of agency – *all things which are the very stuff that maps are made of* – it is difficult not to see those policies as anything but a sanctioned and pre-meditated erasure.



I take the view that the sins of a mother or a father are not those of their sons or their daughters. We have enough evidence to suggest that things never end well when we assume they are. I do not believe however that this allows those same sons and daughters to abdicate their past.

The past I am sorry to say is our burden.

It is our burden because while the stories I've told you today have taken place in a comparatively modest slice of geography to a

comparatively minuscule population *they are the same story everywhere you go*.

The details may change but the story is always the same.

If there is one common thread that binds us all it is that we argue about what to call a place and the meaning of those names.

If there is another common thread it has been to assume the argument is a zero-sum game.

What if we stopped doing that?

I have described to you a series of overlapping grievances and past injustices that *in no way* will a gazetteer remedy alone. A gazetteer is only a tool but perhaps we can stop teaching our tools the bad habits of the past.

A gazetteer is fundamentally about the relationships between one place and all the others. In the past we have been limited by

economics and by physics and by no shortage of myopia in the kinds of relationships a gazetteer might allow.

The hope and the goal for **Who's On First** `https://whosonfirst.mapzen.com` is that the present has afforded us the opportunity to leap-frog many of those limitations. We live in a time when much of the technology around us has taken on a foul-smelling odor but I choose to believe that it is still possible to harness these tools in the service of a common good.

The hope and the goal of Who's On First is not that everyone should have to, or even want to, add their names and stories to our gazetteer but that they *may be able do so* when they choose to.

The hope and the goal is in the chance to allow people the agency to name the places that define their own stories, and in doing so to give them a shared weight and mass in the Universe, without needing to erase someone else's telling in the process.

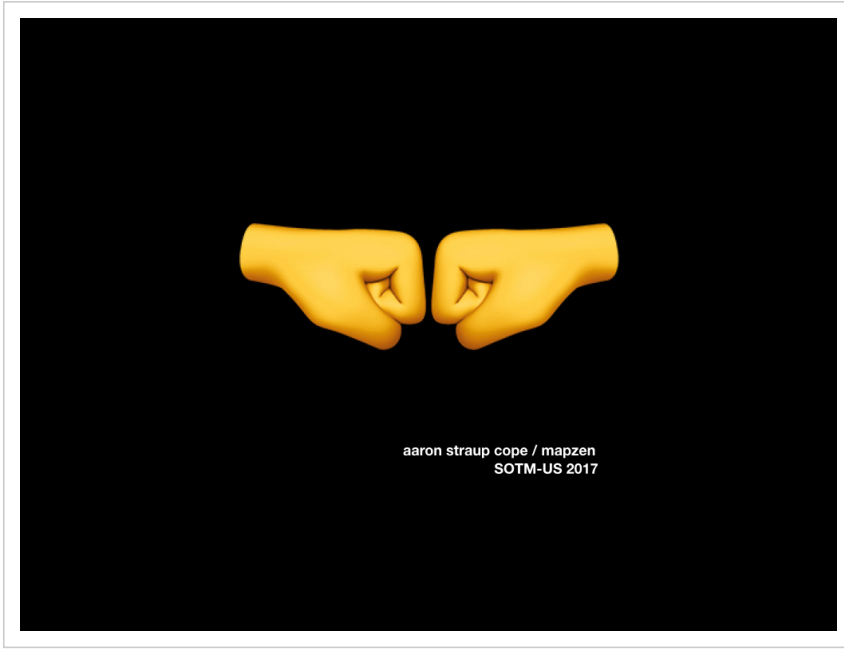
Thank you.

2017-10-17

things I have written about elsewhere #20171024

Who's On First :fist-bump: OpenStreetMap

Who's On First :fist-bump: OpenStreetMap



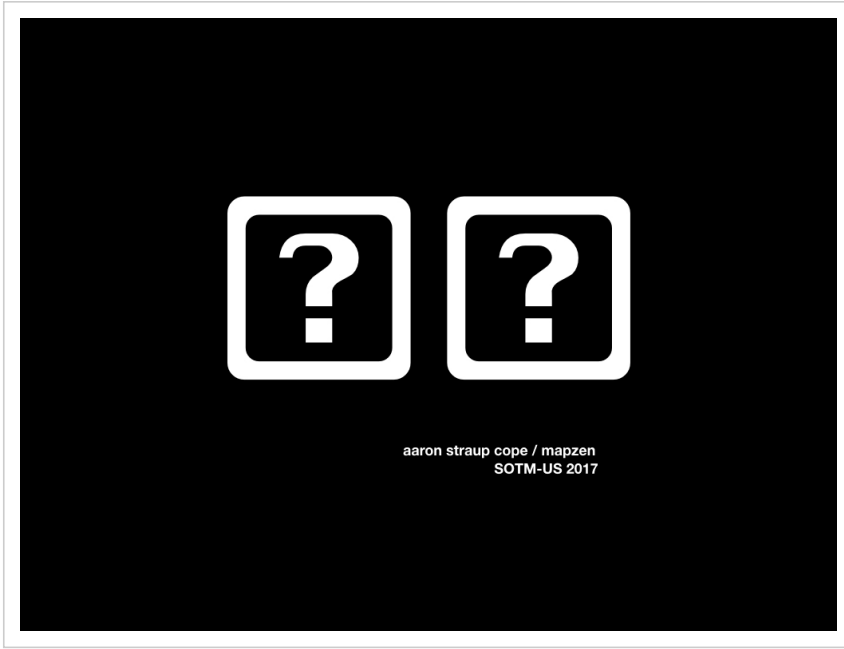
*This was **originally published on the Mapzen weblog*** `https://mapzen.com/blog/whosonfirst-sotmus-2017/` *, in October 2017.*

Hello. I haven't been able to attend **State of the Map** `https://2017.stateofthemap.us` since **2014** `https://2014.stateofthemap.us` so it's good to be back. Thank you for inviting me to speak.

Before we get started I'd like to do a quick bit of house-keeping.

The full title of this talk is "Who's On First colon fist-bump colon OpenStreetMap". The "colon fist-bump colon" part is meant to be a mock **emoji shortcode** `http://www.webstock.org.nz/talks/emoji-fun-profit/` .

There is not an emoji for **"fist-bump"** `https://www.flickr.com/photos/secret_canadian/galleries/72157623155364950/` yet but I'm hoping there will be soon. Now that the **LEFT-FACING FIST** `https://emojipedia.org/left-facing-fist/` and **RIGHT-FACING FIST** `https://emojipedia.org/right-facing-fist/` characters have been added to Unicode 10 all that is necessary to **create a FIST-BUMP emoji** `https://unicode.org/emoji/selection.html` is a combining character.



Not many operating systems have adopted **Unicode 10** `https://www.unicode.org/versions/Unicode10.0.0/` yet, which is why I included a fake emoji shortcode.

For example, that last slide is actually an image. I prepared this talk on a different computer and the laptop I brought to the conference still **can't display Unicode 10 characters** `https://emojipedia.org/unicode-10.0/` .

Tangentially related the software used to publish this blog gets upset when you use the standard ***COLON*** `https://thisisaaronland.github.io/unicode-table/#58` *character in a title which is why we've used the less common* ***MODIFIER LETTER COLON*** `https://thisisaaronland.github.io/unicode-table/#42889` *character in it's place...*

Somewhere between submitting this talk for consideration and its inclusion in the conference program the title was lost in translation.

Instead of newly minted Unicode characters (which would almost certainly **not render** `https://www.flickr.com/photos/bees/9458878023` on a majority of people's screens, never mind the computers used by the printers who made the conference schedule) or my fake emoji shortcode the decision was made to use the more readily available

ONCOMING FIST `https://emojipedia.org/fisted-hand-sign/` character.

I am not really bothered by this except to point out that the title takes on a slightly more threatening and hostile tone than I ever intended.

For the record, punching **OpenSteetMap** `https://openstreetmap.org` (OSM) is not what I had in mind.

This is more what I had in mind.

For anyone who doesn't understand what's going on here, it's two people dressed up as **the Wonder Twins** `https://en.wikipedia.org/wiki/Wonder_Twins` .

The Wonder Twins were Saturday morning cartoon characters, part of the Super Friends alongside more familiar characters like Batman and Wonder Woman.

The Wonder Twins would "activate" their super powers by fist-bumping upon which the siblings would transform in to a pair of complimentary objects. Perhaps **the most famous, and ridiculous, pairing** `https://www.youtube.com/watch?v=FouDjI3GzEI` saw one of the twins become a giant eagle which carried the other who had transformed in to a bucket of water.

The 70s were weird like that in a way that we don't have time to discuss today except to say that Who's On First would like to be the bucket of water to OpenStreetMap's giant eagle.

Who's On First `https://whosonfirst.mapzen.com` (WOF) is a gazetteer. If you've never heard the term "gazetteer" before it's basically just a big phone book, but a phone book of places rather than people. I am going to spend most of this talk discussing venues but I want to start with a high-level overview of what Who's On First is. The shape of the elephant so to speak.

- ***It is an openly licensed*** `https://whosonfirst.mapzen.com/docs/licenses/` ***dataset. At its most restrictive, data is***

published under a Creative Commons By-Attribution license. Whenever possible, and this is true of our own day-to-day work, data is published under a Creative Commons Zero public domain license.

- *Every record in Who's On First has a stable permanent and unique numeric identifier* `https://whosonfirst.mapzen.com/data/principles/` *. There are no semantics encoded in the IDs.*

- *At rest, each record is stored as a plain-text GeoJSON* `https://tools.ietf.org/html/rfc7946` *file. Our goal is to ensure that Who's On First embodies the principals of portability, durability and longevity. This led us to adopt plain-vanilla text files as the base unit of delivery.*

- *Files are stored in a nested hierarchy of directories derived from their IDs.*

- *There are a common set of properties* `https://whosonfirst.mapzen.com/docs/properties/` *applied to all records which may be*

supplemented by an arbitrary number of additional properties specific to that place.

- ***There are a finite number of place types*** `https://github.com/whosonfirst/whosonfirst-placetypes#here-is-a-pretty-picture` ***in Who's On First and all records share a common set of ancestors. As with properties, any given record may have as complex a hierarchy as the circumstances demand*** `https://whosonfirst.mapzen.com/docs/hierarchies/` ***but there is a shared baseline hierarchy across the entire dataset.***

- ***Individual records may have multiple geometries or multiple hierarchies and sometimes both.***

- ***Records may be updated or superseded, cessated or even deprecated. Once a record is created though it can never be removed or replaced.***

- ***Lastly and most importantly Who's On First is meant, by design, to accommodate all of the places.***

Who's On First is not a carefully selected list of important or relevant places. It is not meant to act as the threshold by which

places are measured. Who's On First, instead, is meant to provide the raw material with which a multiplicity of thresholds might be created.

From Who's On First's perspective **the point is not that one place is more important or relevant than any other** `https://mapzen.com/blog/who-s-on-first/` . The point is not that a place may or may not exist anymore or that its legitimacy as a place is disputed. The point is, critically, that people *believe* them to exist or to have existed.

This is why I often refer to Who's On First as **"a gazetteer of consensual hallucinations"** `https://mapzen.com/blog/mapping-with-bias/` .

Here's an example.

This is an animation of **work that my colleague Stephen Epps did** `https://mapzen.com/blog/tackling-space-and-time-in-whosonfirst/` earlier this year to create and link all the records representing the many ways that **the place we used to call "Yugoslavia"** `https://whosonfirst.mapzen.com/spelunker/search/?q=yugoslavia` has changed between the start of the 20th century and now.

I'm going to let this run on a loop while I discuss a point some of you may have tweaked to in the last slide. Specifically that OpenStreetMap and Who's On First have sufficiently different licensing requirements that they prevent data exchange in both directions.

I just said "license" which means somewhere, someone just won **"OSM Bingo"** `https://wiki.openstreetmap.org/wiki/Proposed_features/bingo_hall` .

Seriously though I'd like to make one thing perfectly clear before I go any further: I am not here, either as an individual or an employee of Mapzen, to suggest *in any way* that the OpenStreetMap community revisit **the decision to adopt the ODbL** `https://wiki.openstreetmap.org/wiki/Open_Database_License` .

My own personal feeling is that by virtue of having accomplished the impossible in creating OpenStreetMap, and having done so in thirteen short years, that the community has earned the right to do whatever it wants.

At the same time I still need and want an openly licensed location database that can be used and adapted in both commercial

and "closed" projects without any restrictions beyond attribution.

This is the space that Who's On First occupies and it is why we will not – can not – import data from OpenStreetMap.

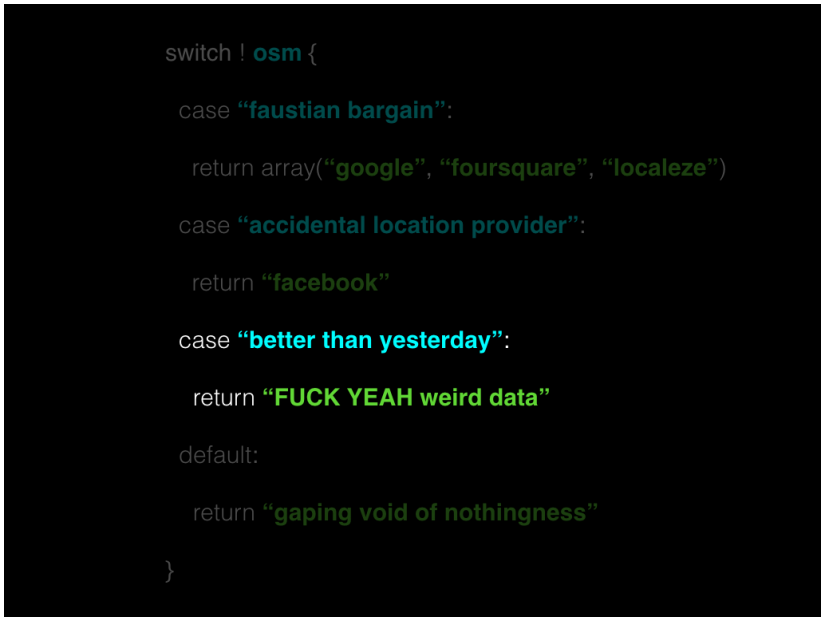
About this time, two years ago, I had a little freak out at work.

We had been discussing the availability and viability of open venue datasets. The reason I freaked out is that when you take OpenStreetMap off the table, for the reasons I've just outlined, there are effectively *no openly licensed venue databases* so I wasn't sure what we were talking about.

Accurate and up-to-date venue data is a difficult and daunting challenge. There are a few companies that have built successful businesses collecting and reselling that data under terms which can only be described as a Faustian bargain.

Other companies like Facebook, we're told, have amassed similar and sometimes superior catalogs of data almost by accident as a by-product of their day-to-day work. So far, though, none of these companies has seen fit to share any of that data.

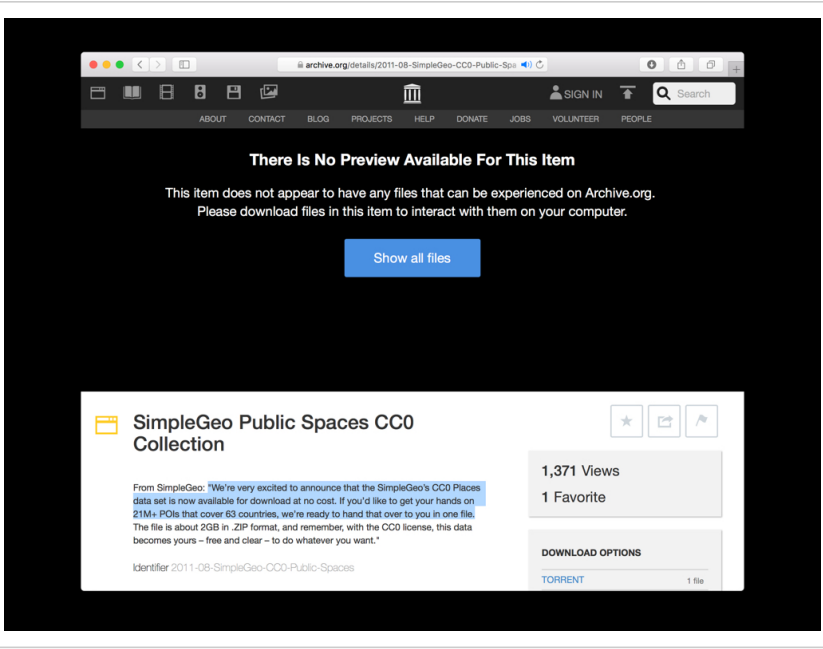
Which really only leaves two alternatives. The first is the gaping void of nothingness that we've all come to know and love. The second is to embrace the idea that *something is better than nothing* and that something which can be improved upon over time is even better than that.



To embrace, or at least accept, that our burden in the near-term will be one of managing absence.

That *in addition* to all the hard work of collecting, vetting and improving data we will also need to think about ways to talk to people about the data we don’t have and to develop interfaces for buffering and tempering that absence.

This makes an already daunting project exhausting even to think about.

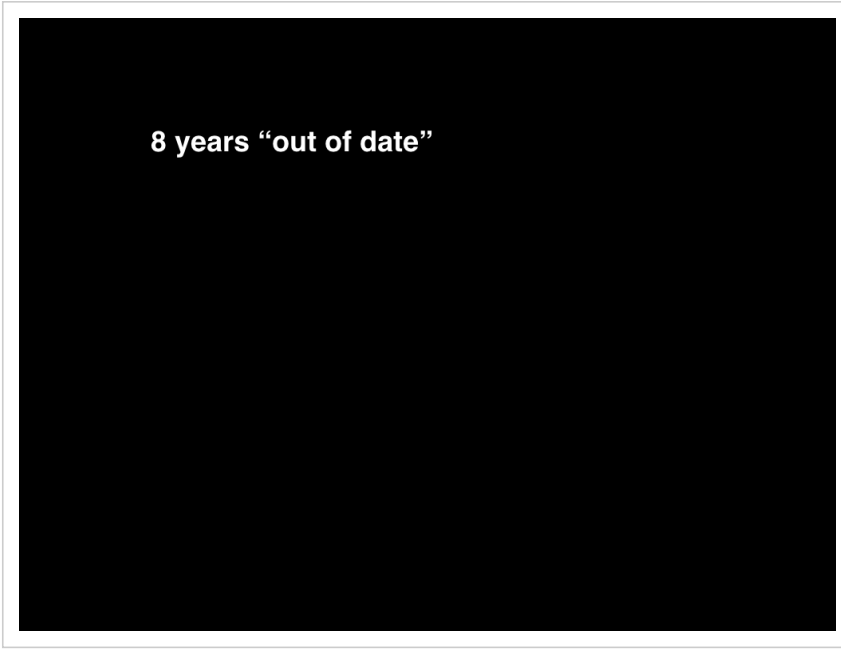


The good news is that there was in fact one open venue database, available for use, in 2015. In 2011 the geo-services company SimpleGeo **published their Places database** `https://web.archive.org/web/20110514083813/http://blog.simplegeo.com/2011/04/20/open-places-data/` , containing 21 million business listings, under a Creative Commons Zero public domain license.

The company went out of business shortly after that but not before **Jason Scott** `https://t.co/1f0N0IPkEt` grabbed a copy of

the data and **put it on the Internet Archive** `https://archive.org/details/2011-08-SimpleGeo-CC0-Public-Spaces` .

The first thing to know about the SimpleGeo data is that it is a flawed dataset.



For starters it is almost 8 years "out of date".

That’s what we say but when we say "out of date" what we are often really saying is there is no *new* stuff.

And when we say "no new stuff" what we’re often really saying is that there is no new stuff for a *particular kind* of business, typically those focused on snacking and grooming and nesting and aimed at a particular demographic: 18-35 year olds, with disposable income.

The funny thing is that lots of businesses survive longer than 17 years. Think of **the butcher that's been in your town** `http://places.mapzen.com/id/572151977` for 75 or 80 years now. If we're lucky, sooner or later we all get older and eventually need to call a plumber or an electrician or a notary.

That's the stuff the SimpleGeo dataset is made of so to say that it's out of date seems inaccurate and unfair.

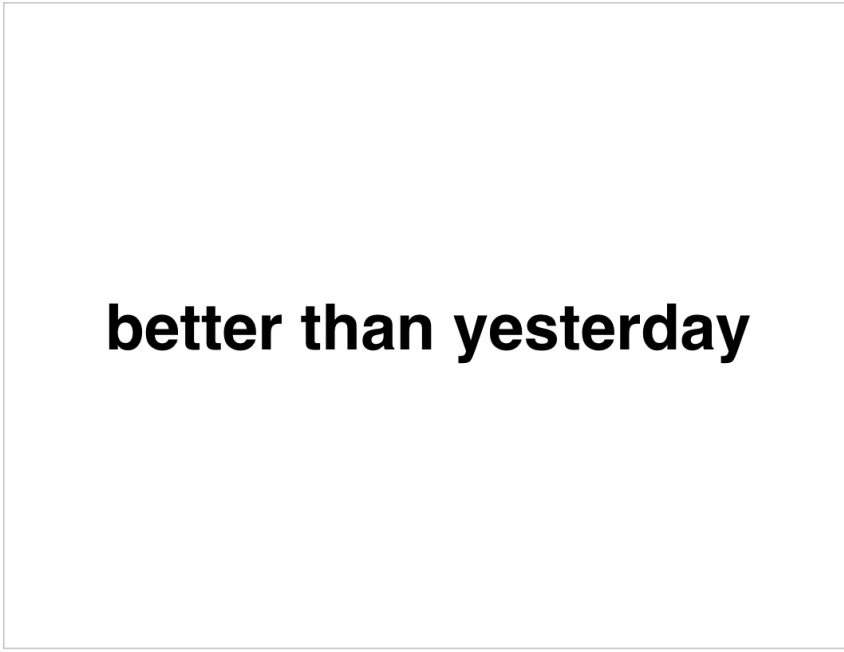
The data is **heavily slanted towards the US** `https://whosonfirst.mapzen.com/spelunker/placetypes/venue/` . SimpleGeo claimed the data covered 60 countries but that is disingenuous when you consider all but about a dozen countries have fewer than a hundred venues.

The data is also heavily weighted towards professional services. There are a lot of CPAs and lawyers and doctors and, to be honest, the sheer volume can make it difficult to see the trees for the forest.

There is some genuinely bad data. No surprises there and I doubt anyone here is ready to cast the first stone when it comes to bad data.

There is minimal structured relational data. For example, the only way to find all the records in a given a locality or neighbourhood is to load everything in to a database and perform spatial queries.

Everything was published as a single line-separated GeoJSON file. Had I been in SimpleGeo’s shoes at the time I might have made the same decision. As a consumer of that data, the decision has been nothing but 100% sad-making.



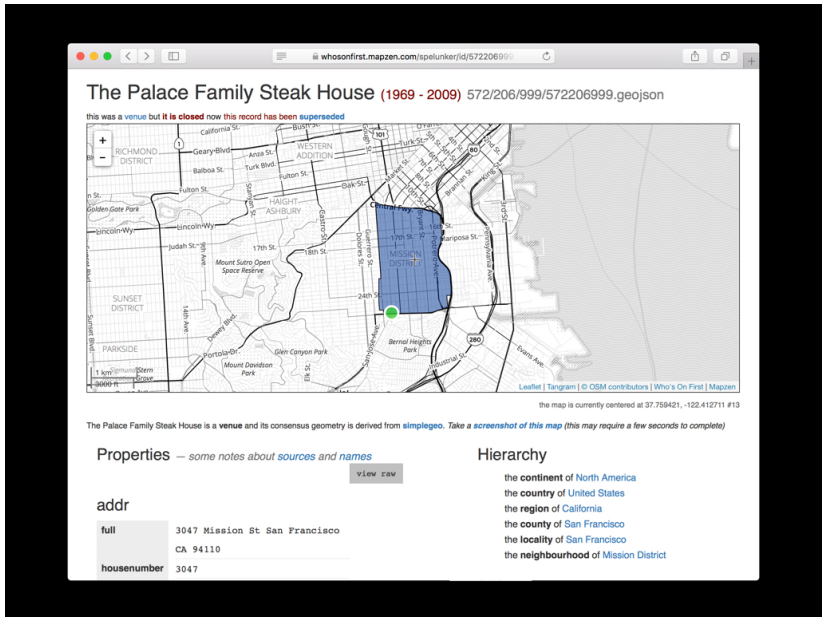
But it's a good start and the first thing we did was to **explode that 21 million line text file in to 21 million discrete records** `https://mapzen.com/blog/whosonfirst-venues` , each with a new Who's On First ID.

If nothing else the data has proven its value in forcing us to think about how, from a purely operational point of view, we manage editing, storing and distributing that much data.

Without getting in to all the details I'll just say that **trying to put 21 million files in to a single Git repository** `https://github.com/whosonfirst-data` remains challenging enough to the point of being impossible.

Simply having 21 million files on a single volume, in 2017, still causes people to **run out of inodes** `https://www.ibm.com/developerworks/aix/library/au-speakingunix14/index.html` and try explaining that to someone who doesn't understand, or care, how filesystems work.

But there are more than 21 million venues in the world and we're going to have to figure these things out eventually so we might as well start now.



This is what that looks like and if all the venues in the world weren't enough, we also want to be able to track historical venues.

Let's pretend that Who's On First existed in 1985 and a writer has published a review of **The Palace Steak House** `http://places.mapzen.com/id/572206999` , associating its Who's On First ID with their article.

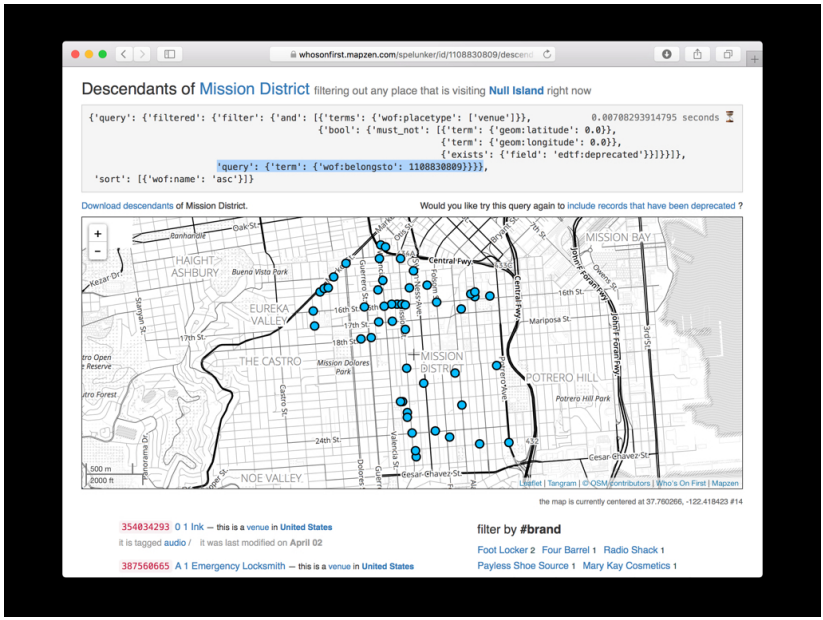
It is important that both the author and the readers of that article can rely on that ID with the confidence that its meaning won't

change even though the Palace went out of business in 2009 and that space has been occupied by

three `https://places.mapzen.com/id/1108800487`

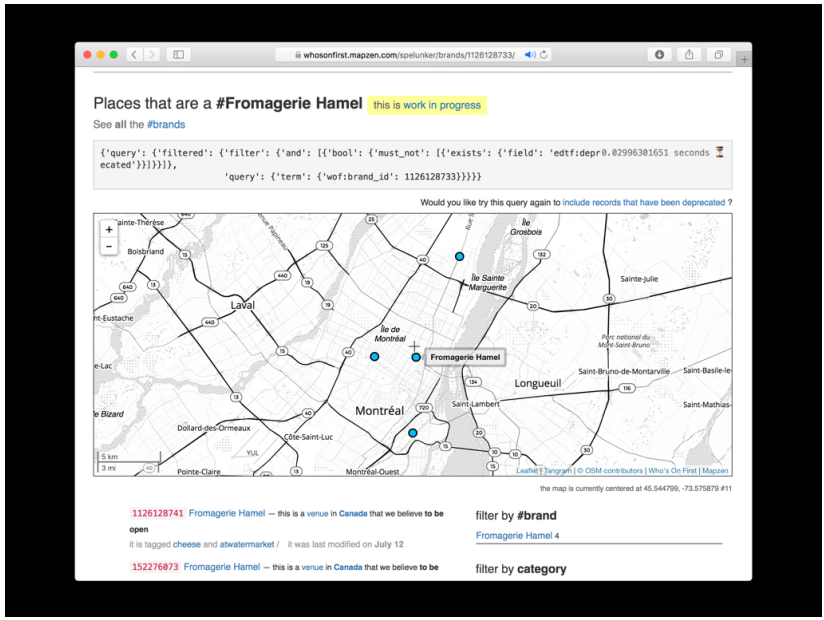
different `https://places.mapzen.com/id/1108800489`

restaurants `https://places.mapzen.com/id/1126130035` since then.



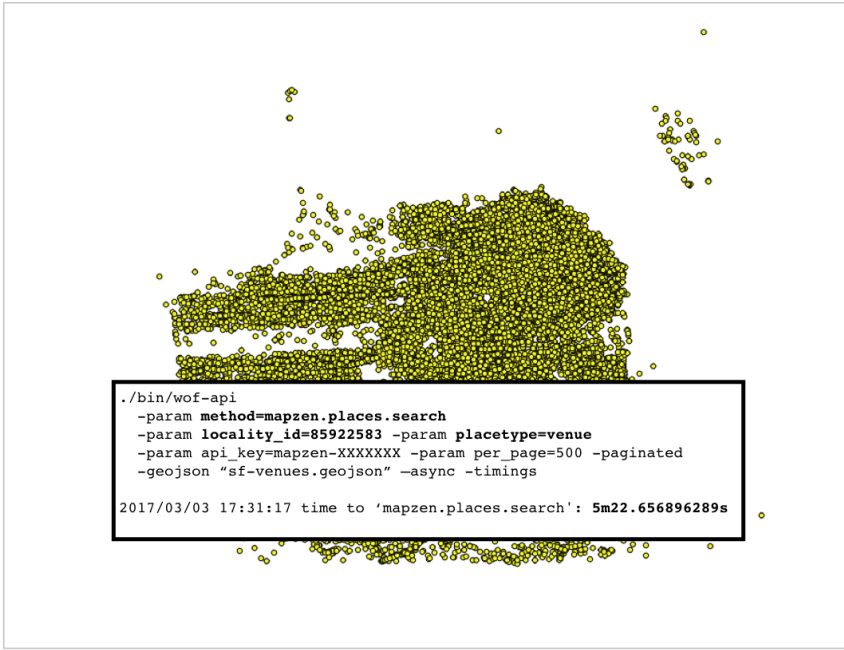
The second thing we did was assign each venue a hierarchy with pointers to all the other places in Who's On First that venue holds hands with.

If nothing else this reduces the problem of **finding all the venues in the Mission** `https://whosonfirst.mapzen.com/spelunker/id/1108830809/descendants/?exclude=nullisland&placetype=venue` to a simple two-part key/value query.



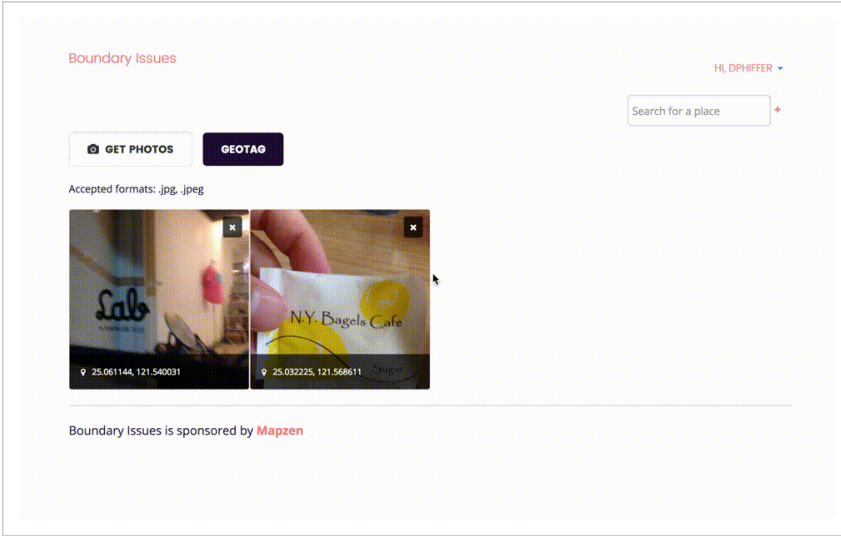
It's early days still but **we are also starting to do the work to track brands** `https://github.com/whosonfirst-data/whosonfirst-brands/` , assigning them each their own stable IDs and static records. It's very primitive work right now and relies entirely on unsophisticated string-matching but it's something to build on.

One of the medium-term goals we have is to work with actual brands and to have them contribute and take responsibility for maintaining their venues inside of Who's On First.



In addition to **static downloads** `https://whosonfirst.mapzen.com/bundles/` everything is available through the **Mapzen Places API** `https://mapzen.com/documentation/places/` .

It takes about 5 minutes to **grab all the venues in San Francisco** `https://github.com/whosonfirst/go-whosonfirst-api` and dump them out as a GeoJSON FeatureCollection.

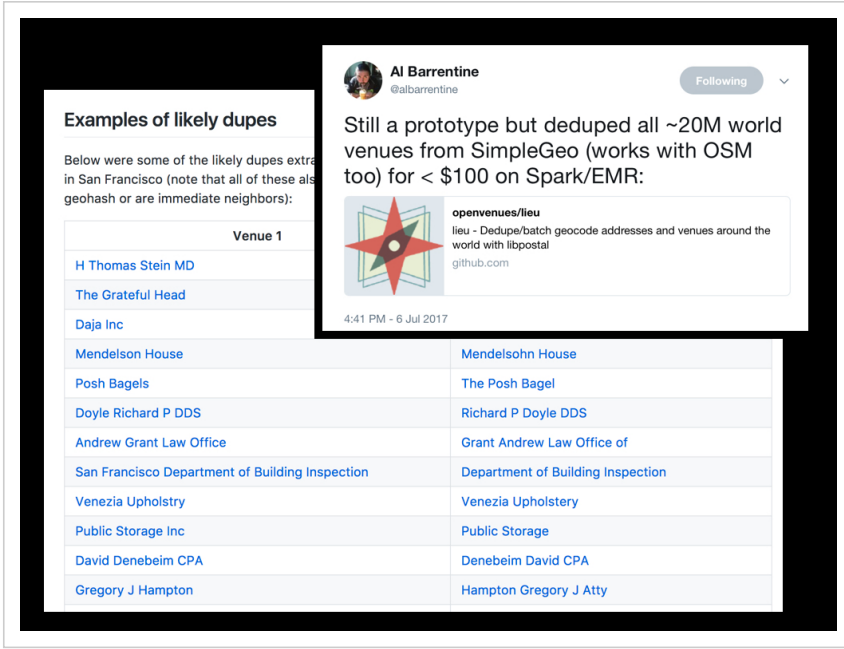


I could have easily spent the entire talk discussing **the UI and UX challenges** `https://mapzen.com/blog/simple-is-hard` around editing the existing data and making it fast and easy to add new data. We are a small team so our biggest challenge is simply the short number of hours in the day.

This is work by another colleague, **Dan Phiffer** `https://phiffer.org/` , to think about how we might **use geotagged photos to add new venues** `https://mapzen.com/blog/geotagging-wof-venues/` .

Dan spent the summer in **South Korea** `https://places.mapzen.com/id/85632231` and **Taiwan** `https://places.mapzen.com/id/85632403`, places where we have (had) no venue data and where all the Mapzen services like tiles and web-based APIs grind to a halt and become unusable when you're traveling on a throttled data plan.

Dan's tool allowed him to roam freely during the day quickly taking pictures of venues and points of interest and uploading them after the fact to create new records in Who's On First.



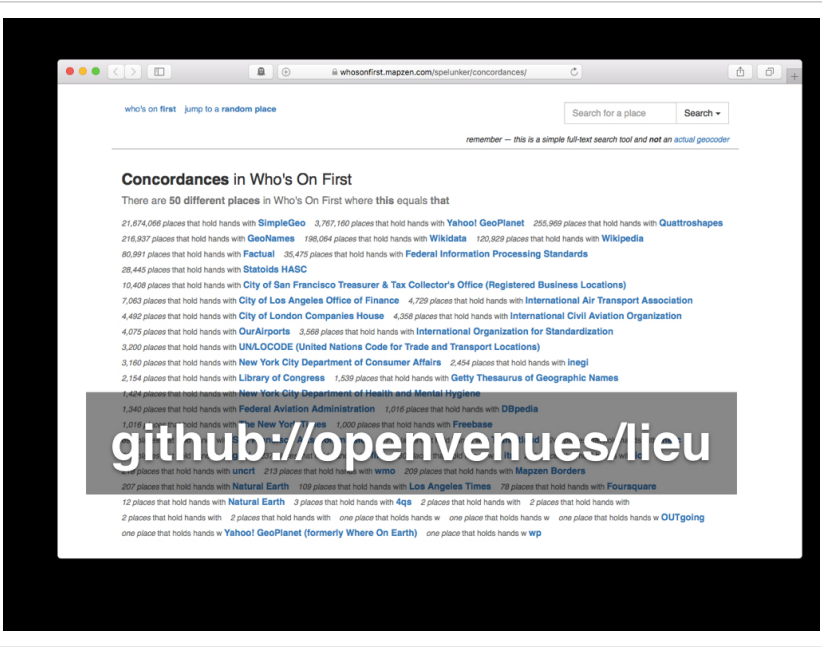
Finally we've been working with **Al Barrentine** `https://twitter.com/albarrentine` , who wrote the **libpostal address parser** `https://mapzen.com/blog/inside-libpostal/` . Building on that work we've asked Al to take a first pass at address de-duplication.

If address parsing is where you go to cry then address de-duplication is where you go to give up.

As we go forward and start to import newer datasets it is clear that the first challenge will be reconciling what we already have in Who's On First with whatever new offering is on the table.

We don't have the choice of giving up on this problem.

Al's remit was not to solve all the things but to build a viable "1.0" tool that demonstrated tangible results and can be improved upon going forward.



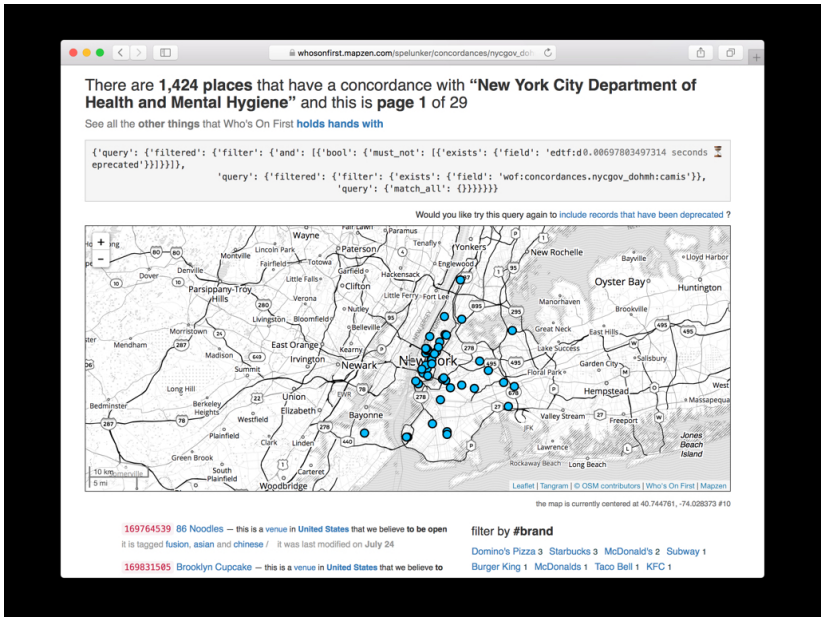
The tool is called `lieu` and, as with libpostal **it is available as open source software** `https://github.com/openvenues/lieu` .

As Al has been working on things I've been testing address de-duplication against Who's On First and publicly available business listings from the usual suspects: **New York** `https://whosonfirst.mapzen.com/spelunker/concordances/nycgov_dca/` , **San Francisco** `https://whosonfirst.mapzen.com/spelunker/concor`

dances/sfgov_rbl/ , **Los Angeles** https://whosonfirst.mapzen.com/spelunker/concordances/lacity_oof/ and **London** https://whosonfirst.mapzen.com/spelunker/concordances/companieshouse/ .

One of the benefits of Al's software is that when we establish a match between of our venues and an official listing we can flag that venue as **"current"** https://whosonfirst.mapzen.com/spelunker/placetypes/venue/?is_current=1 and remove some of the ambiguity that might surround it. We can add **concordances from Who's On First** https://whosonfirst.mapzen.com/spelunker/concordances back to the city and their venue records.

The next step as we get ready to finish up this phase of work and push it out the nest is to de-duplicate Who's On First against itself.



The software is also able to work with fuzzy and approximate location data for venues.

For example, it's possible to use the latitude and longitude for the postal code associated with an address, in the absence of coordinate data for the address itself, and lieu will still find successful matches.

This means that basically **every health and safety inspection record ever published in the**

US `https://whosonfirst.mapzen.com/spelunker/concordances/nycgov_dohmh/` is fair-game for de-duping.

We can validate records that we already have and import new ones with a high, if not perfect, degree of confidence and start chipping away at the "no new stuff" problem.

By now some of you may be wondering what this has to do with **OpenStreetMap** `https://www.openstreetmap.org/` ? The answer is: I don't know but something, I hope.

If Who's On First can't use OpenStreetMap data then we would like to make sure that it's easy for the OpenStreetMap community, if you choose, to use our data. Maybe that means all we do is establish and maintain concordances between the two projects and **maybe it means more** `https://stevecoast.com/2015/09/30/license-ascent/` .

My hope, at the end of this talk, is that I will have raised enough interest and enough questions that we might find out together.

Thank you.

2017-10-24

things I have written about elsewhere #20180102

Who's On First, Chapter Two

Who's On First, Chapter Two

*This was **originally published on the Who's On First weblog*** `https://whosonfirst.org/blog/2018/01/02/chapter-two/` *, in January 2018. I've included it here because it is a fitting end to 2017.*

The short version

We heard you liked 2017 so much that we decided to start 2018 with a bang...

The longer version

So **that's a bummer** `https://www.mapzen.com/blog/shutdown` , yeah?

In many ways everything about the manner in which Who's On First has been designed has been done with this day in mind. We all endeavour to achieve the sort of "escape velocity" that immunizes us from circumstance but that is rare indeed and there was always a chance this day would come. So while "success" was the goal in many ways preventing what I call "the reset to zero" has always been of equal importance.

The "reset to zero" means that one day (like today) the Who's On First you use and pay for as a service on the Internet goes away... and then what?

Typically it's meant that anyone who's wanted to use an alternative to one of the Big Dominant Geo-Services or API Providers (BDGSAP) and has built their applications or services using that alternative has had little choice but to go fishing around in their desk drawer for their BDGSAP account information and think about re-tooling everything... all over again.

That's the "reset to zero".

I would like to think that we have, even just a little bit, prevented this from happening again.

I would be nice to believe that it's only been a reset to "five" (on a scale to ten) but realistically it's probably closer to a reset to "three".

That does not mean that people will be able to turn around and build or offer all of the Who's On First services in a day or even a week but if we've done our jobs well I'd like to think that there is nothing that couldn't be rebuilt (by us or by someone else) in a couple of months. It means that while things are not literally "better than yesterday" – since yesterday you didn't have to read this blog post – it means that things are hopefully better than the yesterday of the last time a service you came to depend on had to shutter its doors.

We'll see, right? The dust hasn't settled yet so there are likely gotchas that remain to be discovered.

The long version

In practical terms what this means for Who's On First in the short-term is:

- ***Both the data and the code will continue to be worked on*** `https://whosonfirst.mapzen.com/spelunker/recent/` ***, but at a slower pace.***

- ***All of the data has been cloned to the Internet Archive*** `https://archive.org/search.php?query=whosonfirst+mapzen` .

- ***All of the code has also been cloned*** `https://archive.org/search.php?query=whosonfirst&sort=-publicdate&and[]=subject%3A%22code%22` ***to the Internet Archive, thanks to the handy iagitup*** `https://github.com/gdamdam/iagitup` ***tool. If you've never donated to the Internet Archive*** `https://archive.org/donate/` ***before this would be a good time to give them a friendly money-hug.***

- ***All of the data*** `http://github.com/whosonfirst-data` ***and code*** `https://github.com/whosonfirst` ***will continue to be hosted in their respective GitHub organizations.***

- ***I have set up an out-of-pocket S3 bucket (and related services) to host a copy of the data, at least in the short-term. This is discussed in detail, below.***

- ***Dan's "Who's On First in a Box" blog posts (parts one*** `https://mapzen.com/blog/wof-in-a-box/` ***and two*** `https://mapzen.com/blog/wof-in-a-box-part2/` ***) demonstrate that it's possible to run most of the WOF stack locally or on an internal network.***

- ***Speaking of blog posts*** `https://mapzen.com/tag/whosonfirst` ***, all the Who's On First related blog posts have been cloned in to the whosonfirst-www*** `https://github.com/whosonfirst/whosonfirst-www` ***repo. They still need some formatting-love but all the text and images*** `https://github.com/whosonfirst/whosonfirst-www/tree/master/blog` ***are safe and sound.***

- ***The current set of editorial tools*** `https://mapzen.com/tag/boundaryissues/` ***will go offline, in the short-term.***

- ***The current version of the API*** `https://mapzen.com/documentation/places` ***will go offline, in the short-term.***

- ***The current version of the Spelunker*** `https://whosonfirst.mapzen.com/spelunker` ***will go offline short-term but I would like to get that up and running somewhere again in the near-term. A side-effect of the API going offline is that the bundler tool*** `https://mapzen.com/blog/bundler/` ***in the Spelunker will go offline until further notice.***

- ***Batch processing and cascading edits will stop in the short-term, pending a review of how and where to spin those back up again. This mostly means time and cost which in a silver-ha-ha-lining kind of way might be a good thing since it's probably time to revisit some of the decisions we've made in the past*** `https://github.com/whosonfirst/go-whosonfirst-updated` ***.***

What's next

A lot of people have been asking me "what's next" and I've generally answered by saying that we'll take some time to lick our wounds (and our bruised egos) and then we'll pick things up again and figure out how to keep it going. But it is work that will have to be done in layers, rebuilding things piece by piece, which is to say:

1. ***Make sure the raw data is available, period. This is mostly done, albeit in human un-friendly forms.***

2. ***Make sure each record is available as an addressable resource (forgive me, I just said "resource"...) on the web. Again, we have one instance of this (`data.whosonfirst.org`) but the more the merrier.***

3. ***Spin up the Spelunker*** `https://github.com/whosonfirst/whosonfirst-www-spelunker` ***again along with its great big Elasticsearch index.***

4. ***Build something akin to the recently announced AWS "S3 Select" API*** `https://aws.amazon.com/about-aws/whats-new/2017/11/amazon-s3-select-is-now-available-in-limited-preview/` ***which allows you to extract individual, specific properties from S3 documents. This is meant to be a bridge between the***

Spelunker and the formal API (below). I have a working prototype for this, today.

5. ***Spin up the API servers*** `https://github.com/whosonfirst/whosonfirst-whosonfirst-api` ***, without any spatial functionality enabled.***

6. ***Spin up the point-in-polygon services*** `https://github.com/whosonfirst/go-whosonfirst-pip-v2` ***and re-enable them in the API.***

7. ***Figure out where and how to rebuild the other spatial queries blah blah blah databases blah blah blah scale blah blah blah computrons blah blah blah... and re-enable them in the API.***

8. ***Rebuild the editorial tools (Boundary Issues*** `https://github.com/whosonfirst/whosonfirst-www-boundaryissues` ***) and figure out how to improve upon the batch/cascading updates. This might also be the right time, or opportunity, to think about how we open up the editorial process more broadly to people on the Internet.***

9. ***It feels like there is enough work in Who's On First, past present and future, to warrant a 3-4 year granted funded project (especially if we aim to make a dent in the "historical places" problem) but that's not anything that will happen right away.***

In my mind everything up to and including the Spelunker is "near-term", the API and the spatial services are "medium-term" and the editorial stuff is "longer-term". It's not ideal but it seems the most realistic given whatever world of new everyone involved in the project will be negotiating during the coming year.

Aside from the near-term work we're doing ourselves (which we'll talk more about below, I promise) we've been in touch with a handful of institutions to see whether they are interested in helping out. In the short term what this has meant is asking whether they would be interested in hosting a static copy of the data (2) and donating a spare computer or two to run the Spelunker and Elasticsearch (3).

Simply hosting the data alone would be fantastic since it would mean that people already relying on existing Who's On First documents will only need to update their URLs to point to `whosonfirst.INSTITUITION.org/data` (or whatever).

Sidenote: In case you've ever wondered why Who's On First defaults to publishing relative URLs for things, this is why...

Running a copy of the Spelunker would be icing on the cake. Everything else seems premature, right now. If you or your institution (or your company) is interested in helping out the dataset consists of 26.5M files and requires approximately 60GB of storage without any Git history data. With Git history data the storage requirements are approximately 300GB. The Elasticsearch index is about 130GB today so let's just round it up to a nice and easy 200GB.

```
./bin/wof-du-stats -mode repo /usr/local/data/whosonfirst-data* | python -mjson.tool
{
    "stats": {
        "0-10k": 26336534,
        "1-10m": 428,
        "10-100k": 170186,
        "10-100m": 30,
        "100-500k": 14563,
        "100m-BIG": 1, <-- oh New Zealand...
        "500k-1m": 827
    },
    "totals": {
        "bytes": 58514102879,
        "files": 26522569
    }
}
```

I tend to think that keeping more copies of things (2,3) in more places is a good thing and also serves as an opportunity to figure out how make them play nicely together across multiple institutions. That's a larger meta-project separate from Who's On First but equally important in its own right.

In the meantime

Here's what we've been doing in the meantime:

- ***https://www.whosonfirst.org*** `https://www.whosonfirst.org` ***is a thing that exists, now. This is the same website*** `https://github.com/whosonfirst/whosonfirst-www` ***that we've always had with a new home. We'll go through and update all the links and pointers in the next week or two.***

- ***https://data.whosonfirst.org*** `https://data.whosonfirst.org` ***and https://dist.whosonfirst.org*** `https://dist.whosonfirst.org` ***are also a thing that exists, too. These are the new homes for Who's On First data. The former is where individual Who's On First records live and the latter is where bundled versions of that data, like the SQLite databases*** `https://www.whosonfirst.org/sqlite/` ***or the original "bundles*** `https://www.whosonfirst.org/bundles/` ***", will live from now on.***

- ***https://id.whosonfirst.org*** `https://id.whosonfirst.org` ***is a first attempt at re-thinking the various "read-only" services to run as containers*** `https://github.com/whosonfirst/go-`

```
whosonfirst-roundhouse#docker .
```

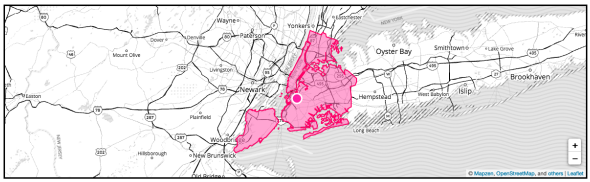
`id.whosonfirst.org` is a simple HTTP-based service that takes a Who's On First ID as its path argument and redirects you to the fully-qualified URL for that record on `data.whosonfirst.org`. For example:

```
curl -s -I https://id.whosonfirst.org/101736545 | grep Location
            Location: https://data.whosonfirst.org/101/736/545/101736545.geojson
```

This is simultaneously an effort to reduce costs, minimize maintenance and overhead and generally make sure that Who's On First plays nicely with all the tools and services. In the same way that Who's On First tries to actively not-care about what sort of database you want to use we try equally to have no opinion about what sort of infrastructure stack you're using. It is early days so it remains to be seen how many of those goals are actually met.

There is also in-progress work to build an **on-demand service for publishing static renderings** `https://github.com/whosonfirst/go-whosonfirst-static` of Who's On First documents. These include HTML, SVG and "standard place response" (or SPR) JSON responses. These static renderings are meant to serve as a bridge between the raw data files and services like the **Spelunker** `https://github.com/whosonfirst/whosonfirst-www-spelunker` or the

API `https://github.com/whosonfirst/whosonfirst-www-api` and live under the `places.whosonfirst.org` domain.



New York is a locality in US . We believe this record to be current.

This record supersedes the following records: ***City of New York*** .

Its Who's On First ID is **85977539** and its relative URI is **859/775/39/85977539.geojson**.

This record's parent ID is `-4` which indicates it has multiple parents because it happens sometimes.

Its bounding box is `40.496133987613, -74.25559136315 2, 40.68295, -73.700009063871` and its principal centroid is `40.68295, -73.9708`.

show raw

abrv

eng_x_preferred	NYC

edtf

cessation	uuuu
inception	1898

geom

area	0.083563
area_square_m	783424470.067211
bbox	-74.2555913632,40.4961339876,-73.7000090639,40.915532777
latitude	40.694457
longitude	-73.930463

gn

elevation	10
latitude	40.71427
longitude	-74.00597
population	8175133

iso

country	US

lbl

latitude	40.68295
longitude	-73.9708

mps

latitude	40.664912
longitude	-73.895946

mz

categories	–
hierarchy_label	1
is_current	1
min_zoom	2

name

afr_x_preferred	New York Stad
als_x_preferred	New York City
amh_x_preferred	ኒው ዮርክ ከተማ
ang_x_preferred	Niweforwīcburg
ara_x_preferred	نيويورك
arc_x_preferred	ܢܝܘ ܝܘܪܟ
arg_x_preferred	Nueva York
arz_x_preferred	نيويورك

status

is current	1	true	true
is ceased	-1	false	false
is deprecated	0	false	true
is superseded	0	false	true
is superseding	1	true	true

hierarchy

continent	102191575
country	85633793
county	Kings County
locality	New York
region	New York
continent	102191575
country	85633793
county	Queens County
locality	New York
region	New York
continent	102191575
country	85633793
county	New York County
locality	New York
region	New York
continent	102191575
country	85633793
county	Bronx County
locality	New York
region	New York
continent	102191575
country	85633793
county	Richmond County
locality	New York
region	New York

sources

https://data.whosonfirst.org/859/775/39/85977539.geojson

https://github.com/whosonfirst-data/whosonfirst-data/blob/master/data/859/775/39/85977539.geojson

For example, the URL for New York City is **https://places.whosonfirst.org/id/85977539** `https://places.whosonfirst.org/id/85977539` and to fetch the SPR response you would do:

```
curl -s https://places.whosonfirst.org/id/85977539.spr | python -mjson.tool
{
    "mz:is_ceased": -1,
    "mz:is_current": 1,
    "mz:is_deprecated": 0,
    "mz:is_superseded": 0,
    "mz:is_superseding": 1,
    "mz:latitude": 40.68295,
    "mz:longitude": -73.9708,
    "mz:max_latitude": 40.915532777005,
    "mz:max_longitude": -73.700009063871,
    "mz:min_latitude": 40.496133987612,
    "mz:min_longitude": -74.255591363152,
    "mz:uri": "https://whosonfirst.mapzen.com/data/859/775/39/85977539.geojson",
    "wof:country": "US",
    "wof:id": 85977539,
    "wof:lastmodified": 1511825453,
    "wof:name": "New York",
    "wof:parent_id": -4,
    "wof:path": "859/775/39/85977539.geojson",
    "wof:placetype": "locality",
    "wof:repo": "whosonfirst-data",
    "wof:superseded_by": [],
    "wof:supersedes": [
        1125397311
    ]
}
```

A few things to note about this example:

1. *It probably makes more sense to use a* `.json` *extension, rather than* `.spr`.

2. *See the* `mz:uri` *property? Depending on when you read this that might have been replaced with a* `wof:uri` *property already. I was (still am) always ambivalent*

*about including that property in the so-called "**standard places response**`https://github.com/whosonfirst/go-whosonfirst-spr#interface`" for just this reason.*

3. *It would be nice to update the* `id.whosonfirst.org` *service to redirect you to the correct HTML, SVG, etc. endpoint based on content headers or equivalent hints.*

I mentioned there are also SVG versions of all the geometries. This is still experimental work so we haven't formalized how it will be operationalized or sorted out all the kinks. My hope is that, in time, we can generate **canned PNG versions from these SVG renderings** `https://github.com/whosonfirst/java-dropwizard-squeegee` for use in applications that just need a picture of a place rather than the raw data.

That's as far as we've gotten, today. As you might imagine it's been a little crazy around here and there's been a lot of other stuff to deal with.

How you can help (with the software)

If you'd like to help out on the software side of things, here's a short list of things (in no particular) order that I've been thinking about:

- ***Document all the things. Really, almost nothing has been properly documented because there have been so few of us and everything's been moving so fast. The bad news is that the documentation isn't great. The good***

news is that if you're feeling like helping out you could probably start with anything in the whosonfirst organization `https://github.com/whosonfirst` *and whatever contribution you make will be an improvement.*

- *Migrate all the py-mapzen-whosonfirst-* *libraries* `https://github.com/whosonfirst?utf8=%E2%9C%93&q=py-mapzen&type=&language=` *from Python 2 to Python 3, ideally with as few changes as possible. The other things I'd like to do is untangle specific libraries that require installing things like* **`shapely`** *(and by extension battleships like* **`gdal`***) for unrelated operations (I'm looking at you py-mapzen-whosonfirst-* *utils* `https://github.com/whosonfirst/py-mapzen-whosonfirst-utils` *).*

- *Speaking of the Python code it's also long past time to document the steps we go through to transform a Who's On First* *document* `https://github.com/whosonfirst/py-mapzen-whosonfirst-search/blob/master/mapzen/whosonfirst/search/__init__.py#L62-L369` *in to something that*

Elasticsearch can index, so that we can index things in other languages.

- *The same is true of the logic for indexing Who's On First documents* `https://github.com/whosonfirst/go-whosonfirst-index` *but this time porting the code from Go to other languages.*

- *Update the Javascript libraries* `https://github.com/whosonfirst/js-mapzen-whosonfirst` *to play well with both nodejs and browser-based applications, alike.*

- *Help improve the tools for validating and repairing geometries* `https://github.com/whosonfirst/java-whosonfirst-jts` *in Who's On First documents.*

- *Backport everything in the whosonfirst-www-api* `https://github.com/whosonfirst/whosonfirst-www-api` *and whosonfirst-www-api* `https://github.com/whosonfirst/whosonfirst-www-boundaryissues` *in to their respective Flamework libraries* `https://github.com/whosonfirst?utf8=%E2%9C%93&q=flamework&type=&language=` *. An*

added bonus would be finally merging the changes back in to the exflickr https://github.com/exflickr *organization but they are so far out of sync at this point that may never happen...*

- *Make Dockerfiles* https://www.docker.com/ *for things where appropriate, like the Spelunker* https://github.com/whosonfirst/whosonfirst-www-spelunker *and the API* https://github.com/whosonfirst/whosonfirst-www-api *and use this as an opportunity to sort out how to manage data bundles, both large and small* https://github.com/whosonfirst-data *in a Docker context. Dan's excellent "WOF in a Box" series (part one* https://mapzen.com/blog/wof-in-a-box *and two* https://mapzen.com/blog/wof-in-a-box-part2/ *) is a good place to start from.*

- *Update the WOF point-in-polygon (PIP) server* https://github.com/whosonfirst/go-whosonfirst-pip-v2 *(which conveniently has a Dockerfile* https://github.com/whosonfirst/go-whosonfirst-pip-v2/blob/master/Dockerfile *already) to optionally work with either Google's S2 geospatial library* https://s2geometry.io/ *or Spatialite* https://www.gaia-

`gis.it/fossil/libspatialite/index` ***in addition to the default RTree implementation. The reasoning here is speed and additional functionality in the case of the former and potentially reducing the memory requirements for the PIP server in the case of the latter. There is an active branch to work on the S2 stuff*** `https://github.com/whosonfirst/go-whosonfirst-pip-v2/blob/s2/index/s2.go` ***although as of this writing it is still returning incorrect results and I haven't had a chance to determine why...***

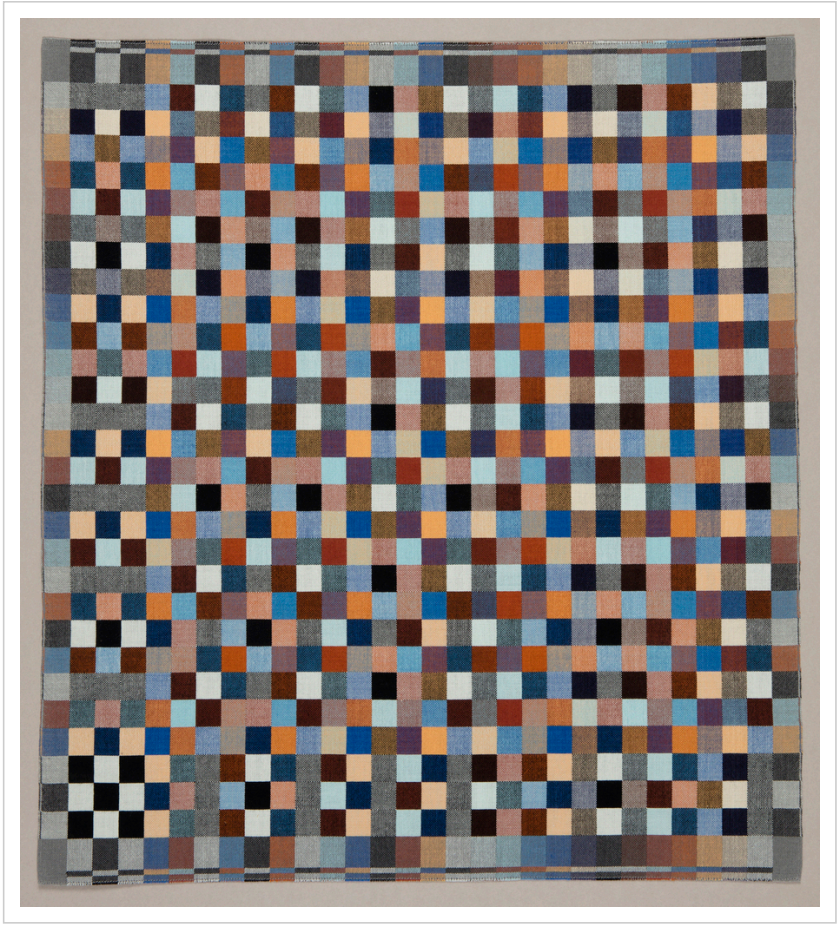
- ***Update the Spelunker*** `https://github.com/whosonfirst/whosonfirst-www-spelunker` ***or the static places.whosonfirst.org pages*** `https://github.com/whosonfirst/go-whosonfirst-static` ***(or both) to optionally read from the Who's On First SQLite databases*** `https://dist.whosonfirst.org/sqlite/` ***instead of Elasticsearch which would allow for limited search and filtering capabilities.***

- ***Finish mapping or importing the remaining Geonames*** `https://geonames.org` ***places used in the MaxMind GeoLite2 city database*** `https://dev.maxmind.com/geoip/geoip2/ge`

`olite2/` ***and that we don't have Who's On First concordances for and then clean up, refactor and document the toolchain we use to build WOF-enabled IP lookup databases and services*** `https://github.com/whosonfirst/go-whosonfirst-mmdb#usage` ***.***

- ***Re-think how changes applied to the "source of truth" for any given record (currently that's GitHub*** `https://github.com/whosonfirst-data` ***roll out to things like places.whosonfirst.org*** `https://places.whosonfirst.org` ***and cause derivative products (like the SQLite databases*** `https://whosonfirst.org/sqlite` ***) to be generated. In the past this was done with a bit of a bubblegum-and-duct-tape solution*** `https://github.com/whosonfirst/go-whosonfirst-updated` ***which got us this far (so that's cool) but is in need of some care and attention.***

How you can help (with the data)



I asked Stephen Epps to outline some ways that people can help out with the data side of things going forward. This is what he said:

Geometry fixes

- *Bulk edits. This includes issues like* ***issue 713*** `https://github.com/whosonfirst-data/whosonfirst-data/issues/713` *, where more than a single geometry is incorrect. These issues are high priority issues because they "trickle down" the placetype chain.*

- *One-offs. These issues are like* ***issue 911*** `https://github.com/whosonfirst-data/whosonfirst-data/issues/911` *and should be taken care of as they arise.*

Data additions

- *These types of issues are data additions for places that Who's On First does not already know about. In some areas of the world, Who's On First just doesn't have data at a specific placetype. See* ***issue 854*** `https://github.com/whosonfirst-data/whosonfirst-data/issues/854` *for an example. This type of issue does not return "wrong" results for a place, but should be fixes so Who's On First has complete administrative data at all appropriate placetypes.*

Property edits

- *Bulk edits. These issues are lower importance and not necessarily an urgent issue for Who's On First.*

- *These issues add Statoids data for "leftovers", update Geonames properties, importing new concordances, and reclassifying name properties.*

- *Property edits can be taken care of in pull requests without point-in-polygon (PIP) work and generally "nice to have" fixes.*

- *One-off. These are the least important, and can be taken care of through some like BI or in a simple PR*

Other issues

- *This includes overall structural issues in Who's On First records, like:* ***issue 975*** `https://github.com/whosonfirst-data/whosonfirst-data/issues/975` *. If you find an issue like the ones listed above in Who's On First or would like to contribute to the project, we will gladly work with you to import recommended changes.*

- *Reviewing the* ***open issues*** `https://github.com/whosonfirst-data/whosonfirst-data/issues` *or filing a new issue*

in the repository is the best way to start. Once we understand the issue, we'll proceed as we have in the past - with a set of property changes, geometry updates, or data additions through a pull request in GitHub.

Tools

- *A useful tool to visualize and categorize these issue would be* ***this one*** `https://github.com/whosonfirst/whosonfirst-placetypes/issues/9` .

To all of that I would add:

- ***Keep telling us where there are mistakes*** `https://github.com/whosonfirst-data/whosonfirst-data/issues` ***and gaps in the coverage. We have our own idea of where there are still problems in the data but the world is a big place and it's likely we've overlooked some things.***

- ***Help us add concordances between Who's On First and all the other gazetteers. Who's On First has never wanted to be the "only" gazetteer but we would like to hand hands with all the other projects out there from***

WikiData to the Getty Thesaurus of Geographic Names to projects we haven't even heard of yet.

- *The same holds true for names (of places). Who's On First aspires to have all the names, and all the nicknames, in all the languages for all the places. We've got a lot of them* `https://mapzen.com/blog/summer-2017-wof/` *but we definitely don't have all of them yet.*

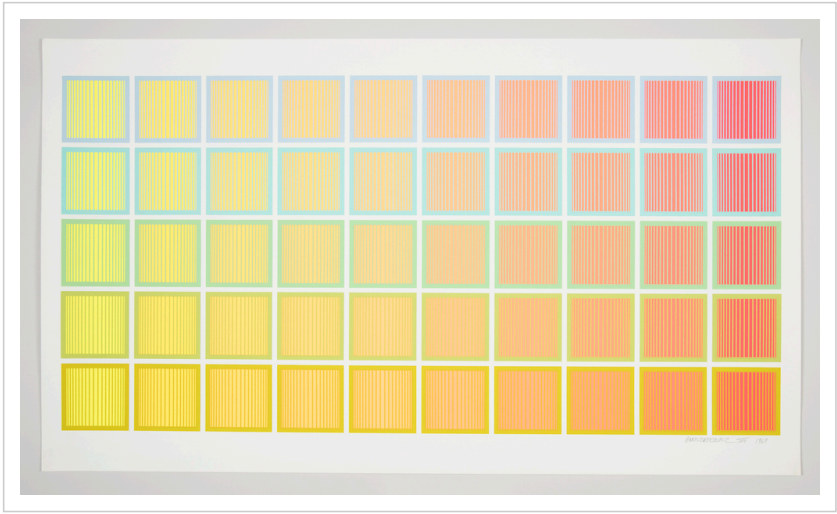
- *Help us start cleaning up the venue data. As with the documentation the bad news is that venues are a big, hairy problem* `https://mapzen.com/blog/whosonfirst-sotmus-2017/` *but the good news is that there are lots of way you can contribute. In the short-term things we're looking for include: Flagging bad or duplicate venues, flagging venues as being "current" or not, adding metadata like addresses or store hours, associating venues with a brand ID* `https://github.com/whosonfirst-data/whosonfirst-brands/` *(which are just like Who's On First IDs but for "brands" which is a term we use loosely to refer to venues that can from one-off locations to giant multinational corporations with stores in every country* `https://github.com/whosonfirst/whosonfir`

st-brands-sizes/tree/master/sizes *). Suggest new venues!*

- *Help find and clean up venues that were initially included in a country-specific repository (because of the ISO country code originally assigned by SimpleGeo, who first published the data* https://mapzen.com/blog/whosonfirst-venues/ *) but whose actual coordinate data put them in an entirely other place.*

- *Help finish up mapping All The Places spiders* https://github.com/alltheplaces/alltheplaces/tree/master/locations/spiders *to Who's On First brands* https://github.com/whosonfirst-data/whosonfirst-brands *. For example, BestBuy* https://github.com/whosonfirst-data/whosonfirst-brands/blob/master/data/420/574/385/420574385.json#L2-L8 *has three spiders:* **`bestbuy`**, **`bestbuy_ca`**, **`bestbuy_mx`**. *Once we have a complete mapping we can better compare Who's On First venues with All The Places* https://mapzen.com/blog/all-the-places/ *venues and import newer data with confidence.*

Inspirational conclusion



It's a disappointing day for sure, but it has been both a luxury and a privilege to work "40 hours a week" on Who's On First for this long. Importantly, things are just a little bit better in January 2018 than they were in July 2015 when we started the project.

We all know that **a comprehensive, high quality and openly licensed gazetteer** `https://www.whosonfirst.org/blog/2016/08/15/mapping-with-bias/` is a need and a benefit to all. Most people, though, have no idea how difficult a problem those three things are to tackle simultaneously and nor should they. There is a lot of work left to do

but my hope is that we have contributed enough to make a dent in the problem at least deep enough for the next person or persons to get a toe-hold and carry things forward still without having to start from scratch.

2017-12-31